NEURAL NETWORK AND RANDOM FOREST BASED EUKARYOTIC GENE PREDICTION METHODS

by

Lonnie Baker

A dissertation submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in
Bioinformatics & Computational Biology

Charlotte

2023

ABSTRACT

LONNIE BAKER. Neural Network and Random Forest Based Eukaryotic Gene Prediction Methods. (Under the direction of DR. DONALD JACOBS)

Gene prediction, a fundamental task in genomics, plays a pivotal role in deciphering the functional elements of genomes. However, accurate gene prediction remains a challenging endeavor, especially when dealing with non-model organisms that often lack extensive phylogenetically similar training data. Current methods which produce highly accurate results on small samples of DNA from model organisms routinely fail to achieve the same level of accuracy in non-model organisms. This dissertation presents two innovative ab-initio gene prediction methods based on neural networks and random forests, addressing the need for improved accuracy and reduced data requirements. Accurate predictions of coding sequences across whole chromosomes will be demonstrated by these methods even when trained on small amounts of data from un-related organisms. Prediction accuracy will also be compared to other leading methods on a highly diverse range of non-model organisms. It will be shown that these methods not only reduce the reliance on extensive training data but also elevate the accuracy of gene prediction, outperforming several existing techniques. As genomics continues to expand its scope to encompass a broader array of species, the contribution of these methods holds promise in advancing our understanding of the genetic landscapes of non-model organisms.

ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. Jacobs for the time he has taken to aid my research over the past few years.

I would also like to thank the members of my dissertation committee, Dr. Fodor, Dr. Guo and Dr. Li for taking the time to learn about my research.

And finally, I would like to thank the members of the UNCC IT team for the amazing jobs they have done maintaining the UNCC computing cluster.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

BA	Balanced accuracy. A measure of prediction accuracy.
F1	F1 score. A measure of prediction accuracy.
NN	Neural network.
PPV	Precision. A measure of prediction accuracy.
RF	Random forest.
SN	Sensitivity. A measure of prediction accuracy.
SP	Specificity. A measure of prediction accuracy.

CHAPTER 1: Dissertation Chapter One

1.0.1 Gene Prediction

The genome of an organism contains a vast amount of information. Its sequence, written in a molecular chain known as deoxyribonucleic acid (DNA), exerts control over every aspect of life from guiding growth to determining behavior and responses to internal and external stimuli.

The development of the Sanger process in 1977 revolutionized the field of biology by allowing the precise sequences of nucleotides in samples of DNA to be determined experimentally. While accurate, the cost of this process was prohibitive in terms of both time and material. In 2005, new methods known as next generation sequencing (NGS) methods were introduced which dramatically reduced the cost and time required to sequence DNA. Sequencing technology has since advanced rapidly[2].

It has long been known that specific regions of DNA known as *genes* play a dominant role in an organisms growth and development. Long before the chemical structure of DNA was determined, it was observed that the inheritance of biological traits followed rules that could be observed but without any clear mechanism of action. The term *gene* was used to describe the as yet unidentified unit by which hereditary traits were passed from parent to offspring. As advances in molecular biology allowed for greater insight, the usage of the word *gene* became more precise.

The most relevant and current definition of a gene is a segment of DNA which serves as a template for the creation of a specific sequence of ribonucleic acid (RNA) through the process of *transcription*. The resulting RNA molecule may code for proteins, in which case it is known as messenger RNA (mRNA), or it may serve a variety of other purposes in which case the transcribed RNA is known as non-coding RNA (ncRNA).

The prevalence of ncRNA in nature and the full range of functions which it performs is currently unknown but is currently a very active area of research[3, 4]. The focus of this work is the identification of genes which code for proteins.

Regions of DNA which code for proteins are known as coding sequences and are usually abbreviated as CDS. In the process of *translation* which follows *transcription*, mRNA serves as a template which guides the assembly of specific amino acids into chains known as peptides or proteins. Given their important role in biological function, identifying the locations and sequences of genes which code for proteins is a critical step in understanding the sequenced genome of an organism. While genes can be located using purely experimental methods, these methods have a high cost in terms of time and resources.

The human genome project (HGP) was a massive research endeavor which had the dual goal of fully sequencing a human genome and locating all genes within that genome. Begun in 1990 and completed in 2004, the HGP required an enormous level of funding and human labor[5]. During the project, many advances were made in sequencing technology and experimental gene locating methods. Despite these advances it was clear that research projects on the scale of the HGP could not be performed for every organism. As the speed and cost of sequencing has improved, experimental methods of gene discovery have indeed proven far too slow to analyze the enormous amount of sequence data being generated. Computational methods have evolved to serve this purpose.

In eukaryotic organisms, genes are composed of alternating sequences of DNA known as exons and introns. Transcription is an enzymatic process by which the exons and introns forming a gene are copied, with modification, into a molecule known as pre-messenger RNA (pre-mRNA). A subsequent process known as RNA splicing removes some combination of transcribed exons and introns from the pre-mRNA to produce a second molecule known as messenger RNA (mRNA). Genes which have only a single exon are a subset of the structure described above where the number of introns is zero. Prokaryotic organisms differ significantly from eukaryotes in that all of their genes are composed of single exons.

As introns are removed from transcribed RNA before the process of translation, only the sequences found in exons encode the information which specifies protein composition. The methods developed in this research focus on the prediction of coding sequences found in the DNA of eukaryotes. These sequences also referred to as coding sequences or with the abbreviation CDS.

The term CDS is often confused with the term exon. The technical difference between these terms is the presence of untranslated regions (UTRs). The first component of a gene is the 5' untranslated region (5'UTR). This region is the location of the initial binding site of the RNA polymerase enzyme and also may contain additional regulatory binding sites. Despite being a part of the completed mRNA strand, UTR sequences are not translated into amino acids in the following translation step.

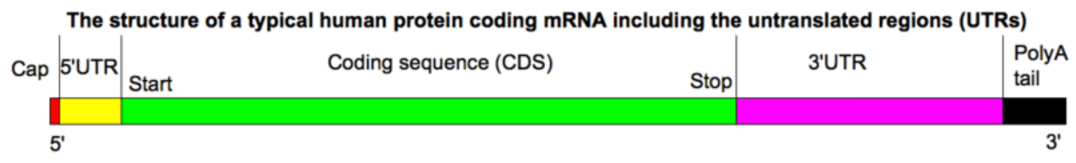

The structure of a typical human protein coding mRNA including the untranslated regions (UTRs)
Cap
5'UTR
Start
Coding sequence (CDS)
Stop
3'UTR
PolyA tail
5'
3'

Figure 1.1: An exon is a coding sequence plus any adjacent UTRs if they are present.

The first *exon* in a gene is composed of a 5'UTR plus an adjacent *CDS*. The final *exon* is a gene is composed of a *CDS* followed by an adjacent 3'UTR. Internal exons which are neither first nor last in a gene are not bounded by UTRs and these internal exons are equivalent to CDS regions. The CDS regions which compose a gene are equivalent to its exons minus any UTRs. This distinction must be clear as the specific goal of this research is to develop methods which can locate CDS regions in unlabeled DNA.

At the end of the 5'UTR which borders the first exon in a gene, a specific codon known as a start codon is located. In virtually all organisms, this codon is represented by the letters ATG. While technically, the term start codon refers to the equivalent translated mRNA sequence AUG, the DNA codon ATG will be referred to as a start codon in this work. This codon also codes for the amino acid methionine in almost all organisms. The start codon marks the beginning of DNA which codes for protein in a particular gene and the codon immediately following the start codon will be the first amino acid in the translated protein.

The process of gene prediction in prokaryotes is much less difficult than in eukaryotes due to the lack of splice sites. When a gene is composed of a single exon, the start codon which bounds the gene must lie within the same reading frame as the stop codon. The term *open reading frame* refers to a reading frame which contains a start and stop codon that are separated by n nucleotides. Where n is an integer multiple of three. In prokaryotes, a gene must always lie inside a single open reading frame. This greatly narrows the search for possible gene locations. This is not true in the case of eukaryotic DNA where start and stop codons may lie in different reading frames prior to RNA splicing. The methods shown here are designed to locate CDS regions in eukaryotic DNA. However, only small modifications would be required to optimize them for prokaryotes.

1.0.2 Homology and ab initio Methods

When a gene is expressed, it must first undergo the process of transcription. This necessarily generates strands of mRNA which can be sequenced. By sequencing mRNA strands, experimental evidence can be collected indicating the precise sequence of a gene. Using sequence alignment methods, this data can be used to locate genes in some circumstances. Beyond several dozen nucleotides, the probability of finding a segment of DNA which perfectly matches an equivalent mRNA strand is extremely low. If such a sequence is found in the genome of an organism, it is therefore very likely that the location of a gene has been found.

In addition to locating genes with sequences identical to known mRNA strands, numerous homology based methods have been developed which attempt to locate genes which have similar sequences to those stored in mRNA sequence databases. These methods include GenomeScan [6], GeneWise [7] and Augustus [8]. Homology based methods can be very accurate in locating genes which are not too dissimilar from known RNA sequences. There are however, many limitations to the use of mRNA sequence data for gene location.

First, a gene which is not being expressed does not generate any mRNA at all. The database of known mRNA stands therefore theoretically has missing data, the amount of which is difficult or impossible to measure. Second, the sequence homology between most genes drops off dramatically when comparing genes from organisms which are not closely related and even in closely relates species the homology can be very low. These factors are significant hindrances to homology based methods. Evidence of this was recently shown when several current homology based methods were tested using the ORForise framework[9].

The ORForise framework is a system of metrics which can be used to assess the accuracy of gene prediction methods in prokaryotes. When compared using this framework, several commonly used homology-based gene prediction methods gave results which were highly unpredictable given the proximity of model and test organisms. In one very clear example, the gene prediction method AUGUSTUS[10] was able to detect 20.53% of genes in the bacterium *S.aureus* when given access to homology data from the *S.aureus* bacterium itself. But when given access to *H. sapiens* homology data, AUGUSTUS was able to detect 78.01% of *S.aureus* genes. While homology based methods were able to make good predictions in some cases, the authors of [9] found that it was not reliably the case regardless of the source of homology data.

A second class of gene prediction methods exist which do not rely on sequence homology at all. Known as *ab initio* methods, they have proven in many situations to be superior to homology based methods. In the study cited above, many of the *ab initio* methods tested were able to locate 99% of *S.aureus* genes. As the accuracy of predictions is compared between homology based and *ab initio* methods on a case by case basis, no one clear winner emerges. As a result, both types of methods continue to be developed.

A wide range of *Ab initio* gene prediction methods exist which are based on statistical measures which differ to some extent between coding and non-coding sequences. Many of these methods are derived from well known problems in the field of signal processing[11, 12]. Measures such as mutual information[13, 14, 15], auto-correlation functions[16], wavelet transforms[17, 18], Fourier transforms[19, 20], digital filters[21, 22, 23, 24], random walks[25], and power-law correlations[26, 27] have proven to have definite correlations with the locations of genes. While the authors of these algorithms always provide some measure of their accuracy, none of them are perfect gene predictors on their own or else the problem of gene prediction would be solved. The methods developed in this research aim to combine multiple *Ab initio* algorithms using machine learning in an attempt to construct a method which has superior accuracy. The feature space in which this method operates is defined by the collection of algorithms employed.

1.0.3 Machine Learning

With the amount of sequence data being generated and the relatively limited number of classes which segments of DNA can belong to, machine learning (ML) algorithms immediately appear suited to the task of gene prediction. Perhaps as a result of this, virtually every type of ML algorithm has been applied to this problem. Examples include neural networks (NNs)[28], random forests (RFs)[29], support vector machines (SVMs)[30, 31], soft computing methods[32] and hidden Markov models (HMMs)[33, 34, 35, 10, 36].

Gene prediction methods based on HMMs are currently perceived as the best methods. This can be seen in comparative studies such as [37] which only compared HMM based methods due to them being the most popular, or the EGASP study in which the highest scores were clearly dominated by HMM based methods[38]. However, the success of neural networks and random forests in many other tasks has inspired this research with a goal of further exploring the potential of NN and RF based methods.

Primitive artificial neural networks can be traced back to the early 1940s when the concept of artificial neurons was first proposed by Warren McCulloch and Walter Pitts[39]. They presented a mathematical model of artificial neurons that could mimic the currently understood basic functions of biological neurons. However, it wasn't until the late 1950s and early 1960s that significant progress was made in the field. Frank Rosenblatt's invention of the perceptron, a single-layer neural network capable of learning and classifying patterns, marked a major milestone[40]. The perceptron demonstrated the potential of neural networks for solving complex problems and sparked widespread interest in the field. Since that time the development of NNs has generally left behind its biological inspirations.

While Neural networks have been applied in many ways to the problem of gene prediction[41, 42, 43, 44, 45], and were considered competitive in the past with methods such as GRAIL[28], they have since been outpaced by hidden Markov model (HMM) based methods in terms of accuracy. As NNs are constantly evolving however, it seems logical to continue research into their usefulness with regards to this problem.

Random forests are a highly developed evolution of simple decision tree methods. The minimal computation time and deterministic behavior of decision trees lend themselves to ensemble learning methods. A random forest is composed of a large number of decision trees which vary from each other in terms of semi-random hyper-parameter permutations. The method was first introduced by Leo Breiman and Adele Cutler in 2001 as an ensemble learning method[46]. The idea behind a

RF is to combine multiple decision trees to create a robust and accurate predictive model. Each decision tree in the forest is constructed using a different random subset of the training data and a random subset of the features. By averaging the predictions of all the trees, RFs can provide more accurate and stable predictions compared to individual decision trees.

The concept of the RF builds upon earlier work on decision trees and bagging. Decision trees are powerful models for classification and regression tasks, but they can suffer from overfitting and instability when applied to complex datasets. Bagging, short for bootstrap aggregating, was proposed as a solution to these issues. Bagging involves creating multiple bootstrap sub-samples from an original training dataset and building decision trees on each sub-sample. A final prediction is obtained by averaging the predictions of all the trees, usually by means of a vote. RFs extend this idea by introducing random feature selection during the tree construction process, which further reduces correlation between the trees and enhances the diversity of the ensemble. This randomness in feature selection and data sampling helps to improve the generalization performance of an RF, making it a popular choice for various machine learning tasks.

As with NNs, RFs have been applied to the problem of gene prediction with some success[25]. As in the case with NNs however, these methods do not currently rank in the top performing methods. As RFs have proven highly successful in other areas, further research into their use in gene prediction seems warranted.

1.0.4 Performance of current methods

While many gene prediction methods report very high levels of accuracy on curated datasets extracted from the genomes of model organisms, accuracy on non-model organisms is far less impressive. This observation has been made by many unbiased researchers and has been noted from the earliest days of computational gene prediction[47] and continues to be made in the current era[37].

A curated dataset is one that is composed of small sub-samples of DNA as opposed to whole chromosomes. A good example is the EGASP dataset[38]. The accuracies of gene prediction methods evaluated on curated datasets from model organisms is often high. For example, the reported nucleotide level accuracies of three top gene prediction methods in terms of sensitivity(Sn) and precision(PPV) on EGASP were: AUGUSTUS Sn= 79% / PPV= 75%, GENEID Sn= 77% / PPV= 76% and GENSCAN Sn= 84% / PPV= 61%.

However, datasets from non-model organisms may serve much better to test gene prediction methods. The creation of such datasets is highly labor intensive partially due to the cost of accurately verifying the location of genes in non-model organisms. Nevertheless, some good curated, non-model organism datasets have been developed. G3PO [37] is a recently developed benchmark dataset which we have used extensively to evaluate the performance of out methods relative to other state of the art methods. When tested against this dataset, the nucleotide level Sn and Sp values for the three methods above were given as: AUGUSTUS Sn= 51% / PPV= 58%, GENEID Sn= 38% / PPV= 52% and GENSCAN Sn= 50% / PPV= 57%. Far lower than what was reported on EGASP. This is not to say these methods are not good. They are in fact considered to be among the top performing methods available.

The true state of gene prediction performance methods as a whole is perhaps debatable due to differences such as the ones illustrated above. In this work we have chosen to assume that state of the art methods in gene prediction perform at the level indicated by results from broad datasets which include both model and non-model organisms such as G3PO. The range of species and extensive testing performed in the construction of this particular dataset make it a logical choice for use as a comparison tool between our method and other current methods.

CHAPTER 2: Research Methods

2.0.1 Method Overview

Two distinct gene prediction methods are described here. The first method is based on neural networks (NN). The starting point for the design of this method was an early neural network based gene prediction method developed by Uberbacher and Mural which was published in 1991[48]. This method was later expanded into the method known as GRAIL[28]. Although GRAIL had early success, it is currently not regarded as being a leading gene prediction method. The NN based method shown here borrows some ideas from [48] related to feature extraction methods while differing significantly in all other aspects.

This method was designed to be trained on DNA from a single organism. When presenting gene prediction results, NNs trained on DNA from different organisms are compared with each other. A primary goal of this method was to develop a prediction method which would require only small amounts of sample DNA as training but would be able to make accurate predictions when shown DNA from organisms which were not necessarily closely related to the training organism. The intended use of this method would involve a researcher selecting a NN which was trained on an organism which was as close as possible to the organism being researched.

The second method is based on random forest (RF) classifiers. This method was designed to combine models trained on DNA from different organisms into a single universal gene prediction method. A user of this method would not be required to select a training organism but would simply apply the fully trained model on whatever organism they wish. While this may sound like a difficult task, the reality is that a significant fraction of organisms currently being studied have no closely related organisms which have been annotated well enough to serve as a closely related training organism.

2.0.2 Feature Extraction

In order to locate CDS regions, this method begins by classifying each nucleotide in a sequence as either coding or non-coding. Coding nucleotides being those which are located inside CDS regions. To do this, each nucleotide must be represented by a vector of numerical features. Feature extraction is the process by which a vector of numerical values is assigned to each nucleotide in a sequence.

Each vector represents a single nt and is a sample which can be used to train or test classification algorithms such as NNs or RFs. The numerical values which are extracted have been selected due to either known or theorized correlations with coding regions of DNA. An example of the feature extraction process for a single nt is given below. Explanation of all features will be given in the following sections.

Suppose that we wish to extract the GC content of DNA as a feature. We begin by selecting a single nucleotide N in the sequence. From the surrounding nts we count the number of Gs and Cs. If, for example we chose to look at a window of 99 nts, this would mean that 44 nts on either side of N would be considered. If 33 of the nts inside this 99nt window were either G or C, the GC content would be $\frac{33}{99} = 0.333$. For the region surrounding nucleotide N, the GC content is 0.333. This number then becomes the value of the GC content feature associated with N. This is what is meant by the extraction of numerical features from DNA. All algorithms which are used to extract features in this research follow this pattern of operation.

For each type of feature that is extracted from a sequence, a different algorithm is employed which extracts numerical values from sequences of DNA. These algorithms have been referred to as *sensors* by some authors such as Uberbacher[48]. Some of the feature extraction(sensor) algorithms below have previously been described in the manuscript[49].

2.0.3 Nucleotide Periodicity

In the process of translation, specific amino acids are added to a peptide chain one at a time. The resulting protein serves a function which is entirely determined by this sequence. The specification of amino acids is stored within a DNA sequence using a system of three letter words known as codons. All amino acids without exception are associated with a specific set of three letters. A result of this is a detectable periodicity of 3 in the occurrence of bases inside coding regions which is not observed in non-coding sequences. Periodicity refers to the occurrence of repeated patterns or cycles at regular intervals within a dataset.

Autocorrelation functions are widely employed in various fields to study the presence of periodicity in data. In the context of time series analysis, autocorrelation functions can identify and quantify the degree of correlation between a sequence and its lagged versions. By calculating autocorrelation functions, researchers can detect and measure the strengths of these periodic patterns. This information is particularly valuable in the field of signal processing, where the identification of periodic signals is crucial for making predictions and extracting meaningful insights.

In DNA analysis, autocorrelation functions have been used to identify periodicity and repetitive patterns within nucleotide sequences. DNA sequences exhibit various levels of periodicity, ranging from short repeats, such as dinucleotide or trinucleotide repeats, to longer repetitive elements, such as transposable elements or gene duplications. By computing the autocorrelation function of a DNA sequence, scientists can uncover the presence of these periodic motifs, which can have significant biological implications. The level of periodicity in DNA sequences can help researchers studying gene regulation, functional elements, and even the evolutionary history of species.

Autocorrelation functions of nucleotide occurrence can be used to measure the period-3 signal which occurs in coding regions. This method was used by Fickett in his gene prediction algorithm[50]. The formula used in this method is borrowed from his work:

$A_1 =$ number of A nts at positions $1, 4, 7...$

$A_2 =$ number of A nts at positions $2, 5, 8...$

$A_3 =$ number of A nts at positions $3, 6, 9...$

$$S^{ac} = A_{autocorrelation} = \frac{Max(A_1, A_2, A_3)}{Min(A_1, A_2, A_3) + 1} \tag{2.1}$$

An alternate method of measuring the period-3 signal of a sequence is a discreet Fourier transform (DFT). Several DFT based gene prediction methods have been developed which aim to utilize this periodicity[11, 24, 51].

To calculate nucleotide periodicity using DFT, the four bases {A,T,G,C} which comprise a sequence must be translated into a mathematical representation using an encoding scheme. Many encoding schemes have been explored in previous research[22, 52, 19]. While some encoding schemes have been shown to result in gene prediction accuracy differences, these differences have always been very minor.

DFT periodicity sensors utilize a binary encoding scheme. DNA sequences are transformed into arrays of binary sequences highlighting the locations of specific bases using the following mapping: $A \to [1000]^T$, $T \to [0100]^T$, $G \to [0010]^T$, $C \to [0001]^T$. Given the stacked binary string $x^{(j)}$ of four rows, the output for the j-th stacked row is given as:

$$S_k^{dft}\left[x^{(j)}\right] = \left| \sum_{n=0}^{W-1} x_n^{(j)}\, e^{-i2\pi n \frac{k}{W}} \right| \tag{2.2}$$

where $x_n^{(j)}$ is the binary value of the element of the string x at position n for stack j. The magnitude of the resulting DFT signal at a frequency of $k = \frac{W}{3}$ is the period-3 signal. The DFT sensor output is given as:

$$S^{dft} = \sum_{j=1}^{4} \left| \sum_{n=0}^{W-1} x_n^{(j)} \, e^{-i\frac{2}{3}\pi n} \right| . \tag{2.3}$$

2.0.4 K-mer Frequency Occurrence

This sensor has been described in previous work[49]. The frequencies of occurrence of specific nt words or k-mers in coding regions are known to be different than those in non-coding regions [53, 31, 54]. Sliding one base at a time within a window, each k-mer is associated with two frequency weight factors, where their ratio is used to differentiate different types of regions. At any location along the sequence, the frequency weight factors of k-mer k inside the coding and non-coding regions are respectively given by $p_1(j|k)$ and by $p_0(j|k)$. Here j runs from 1 to 4^k representing a distinct k-mer. The ratio of these frequency weight factors for the j-th k-mer within a window is given by:

$$S_k(j) = -\ln\left(\frac{p_1(j|k)}{p_0(j|k)}\right) \tag{2.4}$$

For a window of size W nt and a k-mer size of k, a feature is calculated as the sum of all $S_k(j_n)$ where j_n specifies the specific k-mer found along the sequence at position n:

$$S_k^K = \sum_{n=1}^{W-k-1} S_k(j_n) \tag{2.5}$$

where five k-mer sensors are implemented, requiring $k = \{2, 3, 4, 5, 6\}$ to be calculated.

Statistical weights are estimated from DNA sequences in the training sets. To achieve 1000 to 1 statistics, a number of sequences containing a total of $4^k \times k \times 1000$ bases is extracted from both coding and non-coding regions of the training set. The probabilities $p_1(j|k)$ and $p_0(j|k)$ are calculated by the total number of the k-th k-mer divided by the total number of bases within the extracted sequences. When calculating S_k^K of test sequences, the values of $p_1(j|k)$ and $p_0(j|k)$ from the training sequence are used.

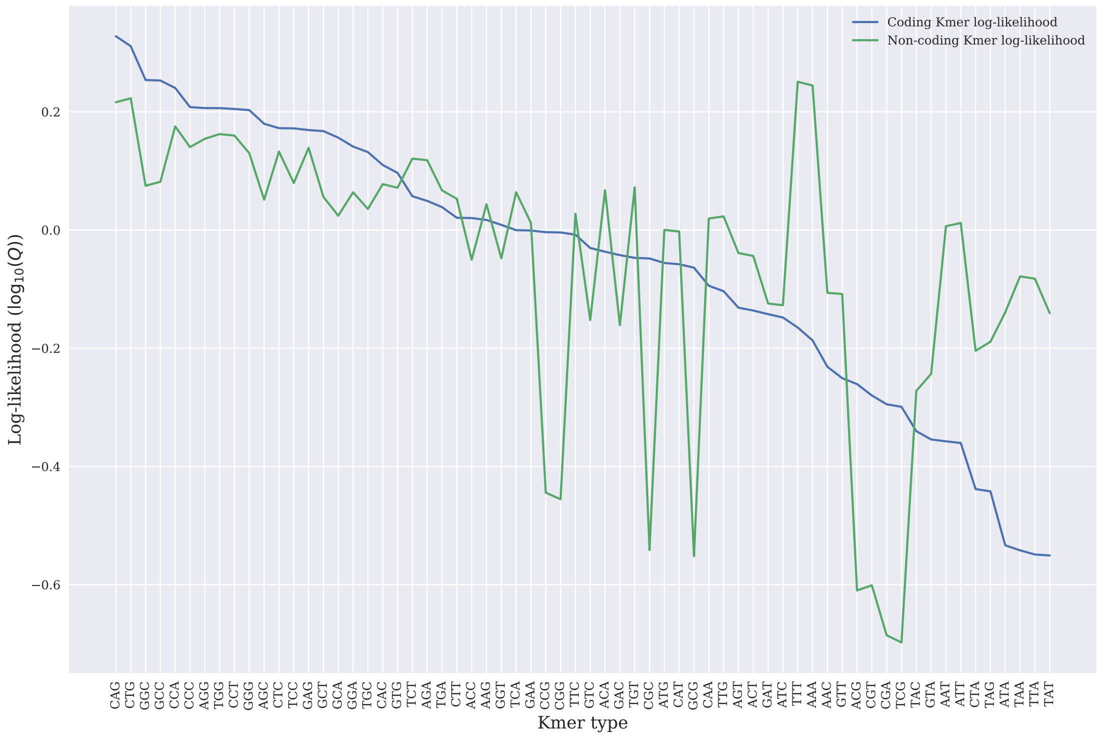


Figure 2.1: An example of the differences in log likelihood of occurances of all kmers of size k=3 in *Drosophila melanogaster* Chromosome 2L.

Greater k-mer sensor effectiveness should occur when k-mer distributions of training and test species genomes are very close. However, even when this is not the case, strong signals can be extracted form k-mer frequency analysis. Evidence of this will be seen in the results section where k-mer sensors are reliably the best predictors of coding nucleotides across many species.

2.0.5 GC Content

The GC fraction of a segment of DNA is known to correlate with many properties of the DNA such as parent species or location in a genome. Higher GC content has also been shown to have a greater propensity for coding regions [55, 56, 30, 41]. The GC content algorithm calculates the GC content of a sequence, denoted as S^{gc}. Let the numbers of each base inside a window be given by n_A, n_T, n_G and n_C, then GC content is given by

$$S^{gc} = \frac{n_G + n_C}{n_A + n_T + n_G + n_C} \ . \tag{2.6}$$

2.0.6 Nucleotide Content

The nucleotide content of a sequence refers to the fraction of the sequence which belongs to a particular nucleotide $\{A, T, G, C\}$. The content of each nt has been shown to differ by a small amount between coding and non-coding sequences[50]. By itself, nucleotide content is only a weak indicator of CDS versus nonCDS status. However, as part of a collection of features is can be shown to make a measurable difference in accuracy[50].

Nucleotide content algorithms calculate the fraction of x nts in a sequence where $x = \{A, T, G, C\}$, denoted as S^x. Let the numbers of each base inside a window be given by n_A, n_T, n_G and n_C, then x content is given by

$$S^{x} = \frac{n_x}{n_A + n_T + n_G + n_C} \ . \tag{2.7}$$

2.0.7 Start and Stop Codon Proximity

This sensor has been described in previous work[49]. Within a segment of DNA containing a gene, start and stop codons are located at natural boundaries between coding and non-coding regions. Moving in the forward direction from 5' to 3', a start codon marks a division point with non-coding sequence to the immediate left and coding sequence to the immediate right. Similarly, a stop codon marks a division point with coding sequence to the left and non-coding sequence to the immediate right. It stands to reason that there could be a statistically significant difference in the sequence composition of DNA which lies to the left or right of these codon motifs.

Four features are defined that take into consideration the nt distance on either side from the query nt to the nearest start and stop codon motifs. For a window size of W nt and distance D nt from the center of the window to the nearest start or stop codon motif, the feature output is given by

$$S^{dis} = \max\left(\frac{W}{2} - D, 0\right) \ . \tag{2.8}$$

Note that the identified motifs within a window are used to determine the distance D, but they are generally not actual start and stop codons. Four start and stop codon features are generated for each sample base: one to indicate the distance to the right of the nearest start codon, one to indicate the distance to the left of the nearest start codon, one to indicate the distance to the right of the nearest stop codon and one to indicate the distance to the left of the nearest stop codon. Furthermore, when no start/stop codon motif is found, $S^{dis} = 0$.

2.0.8 Splice Site Proximity

The same method as the start/stop codon distance algorithm is applied to the nearest canonical splice sites. More than 98% of CDS subregions produced by splicing are bounded by canonical "AG" acceptor sites on their left edges and canonical "GT" donor sites on their right edges. Four algorithms are implemented to track the left and right sides of the “AG” acceptor site and the “GT” donor site.

2.0.9 Entropy

This sensor has been described in previous work[49]. It has been shown that coding regions have less entropy than non-coding regions [57, 58, 59]. A window of nucleotides can be viewed as an emitter for a random sequence. Neglecting correlations, the naive entropy per base of this emitter is calculated as

$$S^{ent} = \frac{-1}{W} \sum_{b} n_b \, P(b) \log P(b) \tag{2.9}$$

where $P(b)$ is the probability of base b with $b = A, T, G, C$. The values of $P(b)$ are calculated by counting the occurrence of each base inside the window. For example, $P(A) = n_A/W$. Unlike k-mer sensors, the statistical weights for $P(b)$ are determined based on the numbers of nucleotides observed within the window.

2.0.10 Classification Methods

The sections above define the feature space used in this method. Below is a description of the two ML methods that operate inside this space. A brief overview of some basic hyper-parameters used by the NNs and RFs will be given here. An advantage of this method is its modular design such that changes to NN or RF configuration can be implemented independently of modifications to the sensors or feature space.

2.0.11 Neural Network Classifiers

All neural networks were implemented in TensorFlow. The results presented here all feature dense, fully connected architectures. All neuron use Scaled Exponential Linear Unit (SELU) activation functions[60]. This activation function has proven highly successful in other fields an some initial experimentation showed that it gave better results than other activation functions although a highly detailed analysis of alternate activation functions was not conducted. Training loss functions were determined using sparse categorical cross entropy from logits. All NNs were trained using the Adam optimization algorithm[61].

Five fold cross validation was implemented by dividing each training set into fifths and training five separate NNs, each on a different 80% subset of the total training set. In each case the 20% of training data left out served as a validation set. All NNs were trained using 200 epochs although a final stable value of the validation loss function was always observed very early in training. After many tests, no additional gain was ever observed in terms of validation loss functions beyond about 100 epochs depending on the parameters of the model being trained. The choice of 200 training epochs more than exceeds the point of zero gain in all cases.

2.0.12 Random Forest Classifiers

Random forests were implemented in R using the randomForest() package. The number of variables RFs were trained by minimization of the Gini impurity index at each branch:

$$G_i = \sum_{i=1}^{N} p(i) * (1 - p(i))$$

where N is the number of possible classes and $p(i)$ is the fraction of that class in the predictions as each branch. A branch which has only members of a single class would have a Gini impurity of zero because for the given class $p(i) = 1$ and $1 - p(i) = 0$ while for all other classes $p(i) = 0$. The branch then has a 'pure' separation of classes and would allow for perfect classification of samples which arrive on that branch of the decision tree.

Five fold cross validation was not implemented for RF methods here. Many training runs using different sets of randomly selected nucleotides quickly showed what was also observed with the NN based method. When fully trained, RFs with identical architectures would always converge to virtually the same accuracy values on a given validation set. For this reason, it was deemed unnecessary to implement five fold cross validation.

One advantage of RF methods is their ability to clearly illuminate the importance of each feature in the sample space. Each time a branch is created within a decision tree, a value is determined which splits the data into two categories based on a single feature and minimizes the Gini impurity index of both branches combined. If, over the course of training a large number of decision trees, a particular feature is shown to have very little impact on reducing the Gini index, that feature can be considered less important. This feature of RFs was used to adjust the types of sensors in the final models due to certain sensors low contribution to accuracy.

2.0.13 CDS Prediction Algorithm

In eukaryotic organisms, with few exceptions, the CDS regions that make up genes are bounded by highly conserved motifs. Vast amounts of experimental evidence has been collected which indicates that the identities of nucleotides immediately adjacent to a CDS region are not free to take on any values. Currently this method only takes into consideration canonical start, stop and spice sites. Many studies have been conducted which show that these sites account for over 98% of all sites [62, 63, 64].

2.0.14 Determination of potential CDS regions

A description of this process has been published in[49]. A gene composed of a single CDS begins with an ATG start codon followed by an integer number of three letter codons, and ends with one of the three canonical stop codons, {TAG, TAA, TGA }. All prokaryotic genes belong to this single CDS form. The genes of eukaryotic organisms are usually composed of several CDS regions, which are combined by splicing before translation. A multiple CDS gene consists of three types of CDS regions, referred to as "start," "internal" and "stop" CDS regions. A "start" CDS begins with an ATG start codon, followed by a number of bases and usually terminating in a canonical GT donor splice site. A typical "internal" CDS is bounded by the canonical AG acceptor and GT donor splice sites. An "end" CDS begins with an AG acceptor site and terminates with a stop codon.

A search for all start (ATG) and acceptor (YAG) site motifs is conducted across the entire sequence. When a start or acceptor motif is found, a secondary search for downstream stop {TAG, TAA, TGA } or donor (GT) motifs is conducted. When a downstream stop/donor motif is found, the region between the start/acceptor and stop/donor is considered a potential CDS and the site coordinates are added to a list of all potential CDS regions. The search for downstream stop or donor motif sites continues until a user defined number of motif sites have been passed. This number serves to limit the maximum length of potential CDS regions.

Limiting this length is mostly done to reduce computing time as this process generates a very large number of potential CDS regions through combinatorics and extremely long CDS regions become extremely rare past a certain point. Studies of exon length distributions across many species have been conducted such as this one by Hawkins[65]. These studies in combination with potential user knowledge of individual organisms is used to determine a good estimate for maximum CDS consideration length.

An earlier implementation of this method used a set number of nts to limit maximum CDS consideration length. However it was found that counting downstream stop/donor motifs was consistently more accurate(see results). An alternative method was proposed which would set a limit on CDS lengths by counting the number of accepetor (YAG) sites instead of donor (GT) sites. However, the total number of YAG sites was extremely close to the total number of GT sites for any given CDS length (figure 2.3).

The number of stop or donor motifs which are found within CDS regions but are not true splice sites was found to be species dependent. For example, in textitD. melanogaster chromosome 2L, slightly more than 5% of CDS regions contained more than 50 GT sites which were not true splice sites but in textitH. sapiens chromosome 21 less than 0.6% of regions contained this many non-splice GT sites. This observation is related to the known differences in mean CDS length between species. The number of stop/donor sites to consider which lie downstream from each start/acceptor site is a user defined parameter. In practice, an accurate setting is not required as no difference in accuracy is observed across a very wide range of values once the total

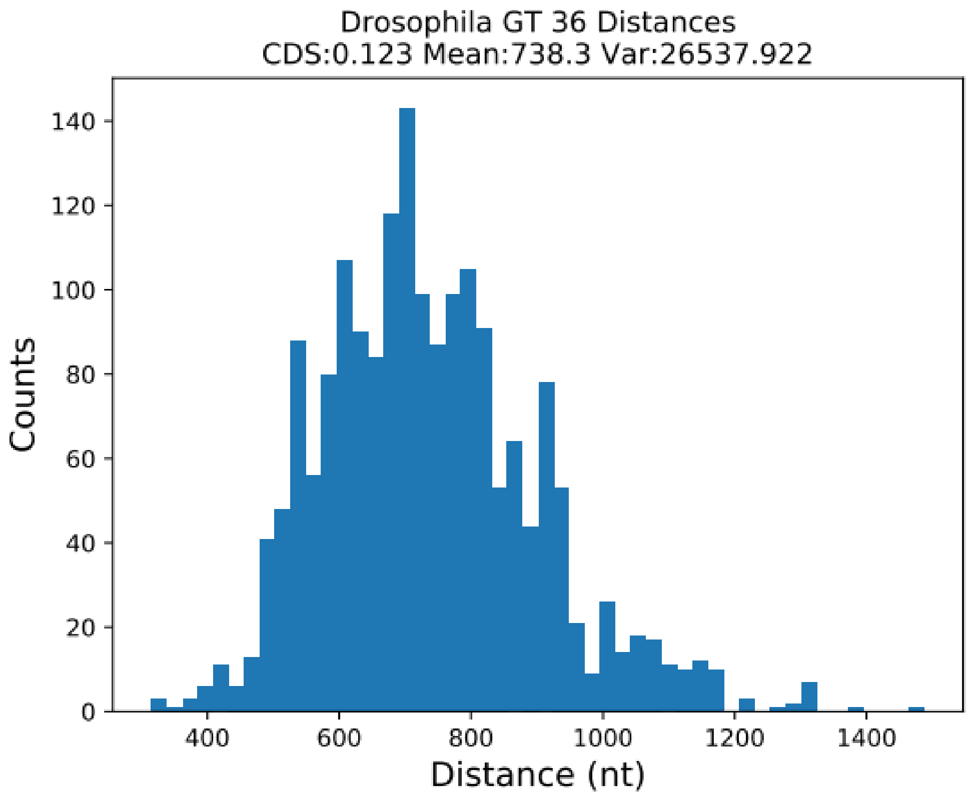


Figure 2.2: A distribution of CDS lengths for all CDS regions in textitD. melanogaster chromosome 2L which contained 36 or more GT motifs.

number of sites to consider exceeds about 30 GT or stop motifs. This make sense as limiting the maximum length too much would prevent the method from locating long CDS regions while overestimating the limit poses no inherent problem beyond increasing computation time.

2.0.15 CDS/nonCDS voting threshold

A description of this process has been published in[49]. Within each potential CDS region, the *initial predictions* are counted. If $N1$ equals the number of positive initial predictions (coding nts) inside the potential CDS and $N0$ equals the number

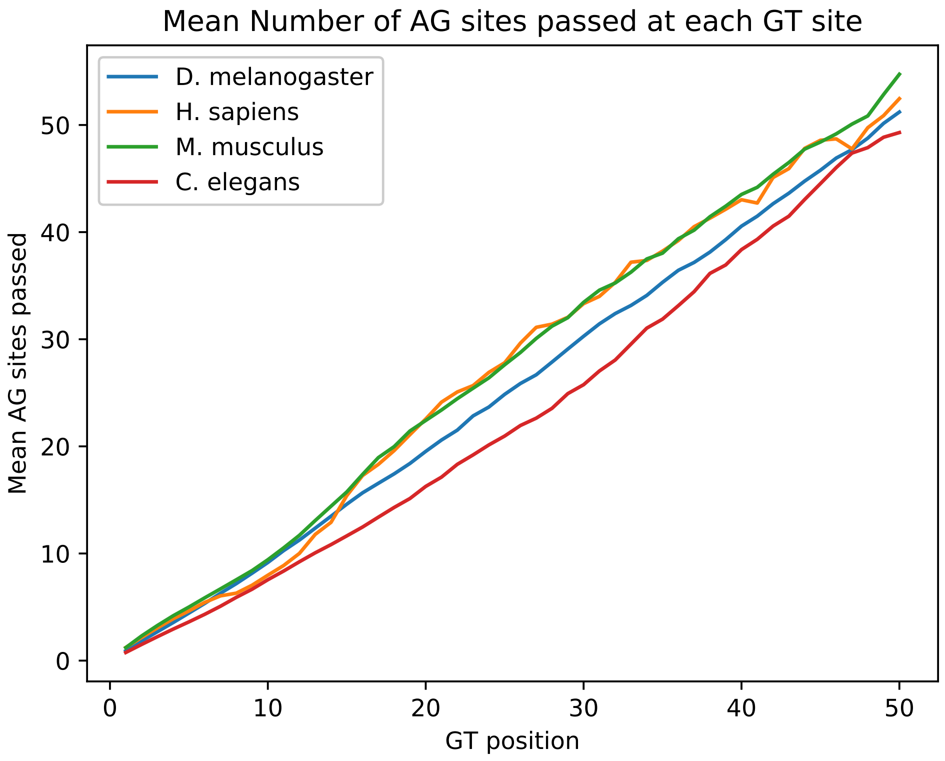
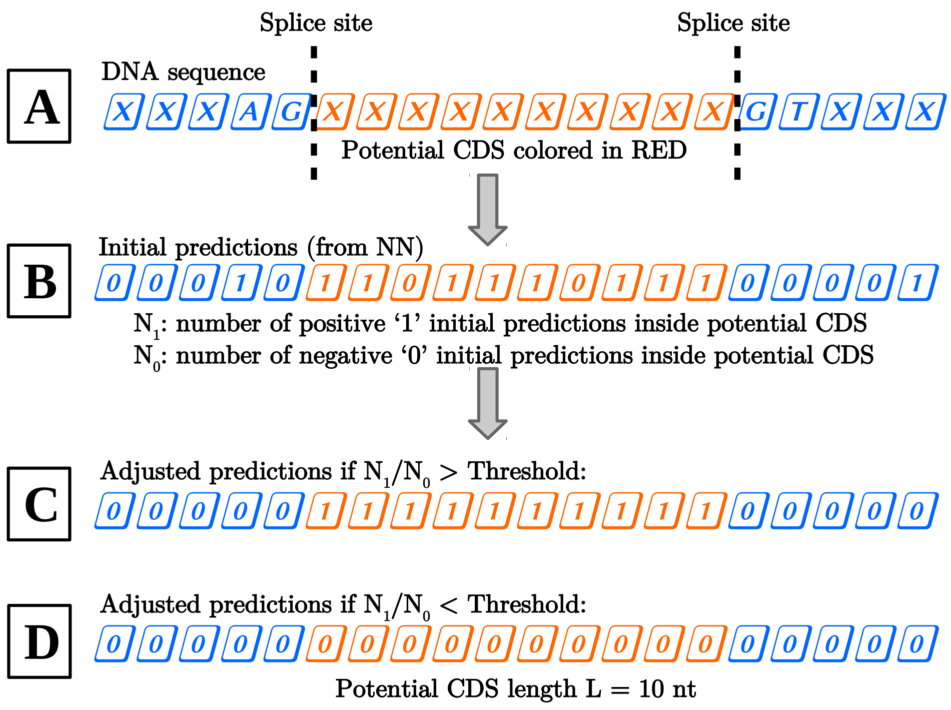


Figure 2.3: The number of YAG acceptor sites passed inside all CDS regions is strongly correlated wit the number of GT sites passed. This plot represents combined data from all model organisms used in this study.

of negative initial predictions then a ratio $\frac{N1}{N0}$ can be assigned to every potential CDS region. To generate a set of adjusted predictions, a user defined CDS/nonCDS threshold is chosen. All potential CDS regions for which $\frac{N1}{N0}$ is above this threshold are adjusted so that all nts within the potential CDS are converted to positive predictions. Potential CDS regions for which $\frac{N1}{N0}$ is below the threshold are converted to zeros. All predictions for regions which lie outside a potential CDS are also converted to zeros. In other words, a potential CDS is predicted to be a true CDS if the following condition is met inside the CDS region:

$$\frac{Positive\ predictions}{Negative\ predictions} > Threshold$$



Splice site
Splice site
A
DNA sequence
X X X A G X X X X X X X X X X G T X X X
Potential CDS colored in RED
B
Initial predictions (from NN)
0 0 0 1 0 1 1 0 1 1 1 0 1 1 1 0 0 0 0 1
N_1: number of positive '1' initial predictions inside potential CDS
N_0: number of negative '0' initial predictions inside potential CDS
C
Adjusted predictions if N_1/N_0 > Threshold:
0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0
D
Adjusted predictions if N_1/N_0 < Threshold:
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Potential CDS length L = 10 nt

Figure 2.4: CDS prediction algorithm. **[A]** The location of splice site motifs indicates a potential CDS. **[B]** Initial NN predictions classify each nt as either coding 1 or non-coding 0. Within a potential CDS, the number of initial coding predictions is N1 and the number of non-coding predictions is N0. **[C]** If $\frac{N1}{N0}$ is above the current CDS/nonCDS threshold, all nts inside the potential CDS become positive 1 predictions. **[D]** If $\frac{N1}{N0}$ is below the threshold, all nts inside the potential CDS become negative 0 predictions. Nts which do not lie between valid splice site motifs are automatically converted to negative 0 predictions.

The CDS/nonCDS threshold affects sensitivity and specificity. As the CDS/nonCDS threshold increases, the number of positive *initial predictions* inside a potential CDS required to mark that CDS as true increases and fewer potential CDS regions will meet the threshold. This causes the number of false positives to decrease but unfortunately will also inevitably reduce the number of true positives as some true CDS regions may not meet the higher threshold. The converse applies to both true and false negtives with both decreasing as the threshold increases. The result of this is a decrease in sensitivity and increase in specificity at higher thresholds.

The trade off between sensitivity and specificity is not always equal however and there is always a range of CDS/nonCDS thresholds which give superior results in terms of balanced accuracy (BA). Balanced accuracy is the mean of sensitivity and specificity:

$$BA = \frac{Sn + Sp}{2} \tag{2.10}$$

As a single metric, BA gives an incomplete picture of how good a set of predictions is. However it has proven a useful way of comparing results when increases in Sn are accompanied by decreases in Sp or the reverse.

2.0.16 Setting the CDS/nonCDS voting threshold

Calculating Sn and Sp from predictions on a known dataset gives insight into how the CDS/nonCDS threshold behaves. However, when attempting to locate genes in a new organism it is impossible to calculate these numbers. A system of setting the optimal CDS/nonCDS threshold without access to known labels must therefore be employed. One statistic that can always be calculated without any knowledge of the species being studied is the total fraction of DNA which was predicted to be coding by the method.

The fraction of DNA which codes for proteins in the genomes of many organisms can be estimated from similar organisms. In humans, a mean coding fraction of 2.07% has been estimated for the whole genome[1]. Variation exists between chromosomes. Coding fractions of chromosomes 18, 19, 21 and 22 are 1.2%, 6.2%, 1.2% and 2.5% respectively. Most other mammals also have a similarly low coding faction of their genome. For example, estimates of some mammalian species fractions of coding DNA are given by:*H. sapiens*: 2.07%, *M. musculus*: 2.55% and *Pan troglodytes* 2.27%. This contrasts to estimates of the following single celled yeast organisms: *Saccharomyces cerevisiae*: 72.45%, *Candida glabrata*: 64.95% and *Schizosaccharomyces pombe* 57.70%.

Our results have shown that the predicted fraction of coding DNA can be used to estimate an optimal CDS/nonCDS threshold without any reference to labeled data. A sweep over CDS/nonCDS thresholds is carried out, starting low and gradually increasing. At each iteration, all potential CDS regions are evaluated as either true or false. The total fraction of test DNA predicted as coding is then calculated by dividing the total number of nucleotides inside true CDS regions and dividing by the total number of nucleotides in the test sequence. If this fraction is significantly above the expected level, the CDS/nonCDS threshold is increased again and the process repeats. Once a set of CDS predictions are made for which the fraction of the predicted coding nucleotides is below a user defined estimate, the process is complete and the current predictions are accepted.

For example, if the fraction of coding DNA on *H. sapiens* chromosome 21 was predicted to be above 0.25 or 25%, it would be obvious that the method is generating far too many false positives. By increasing the CDS/nonCDS threshold, a higher level of evidence is required before predicting each CDS region. This results in fewer predicted CDS regions and a lower predicted fraction of coding DNA. When predicting CDS regions in *H. sapiens* chromosome 21, the CDS/nonCDS threshold was increased until less than 10% of the nts in the chromosome where predicted as coding.

After this threshold is set, Sn, Sp and other metrics are calculated. Plots of accuracy metrics as functions of threshold value or predicted coding fraction will be shown in the results section which will give insight into the threshold setting process. More examples of settings currently used in this research include the fruit fly *D. melanogaster* and flatworm *C. elegans* with coding fractions below 0.17 and 0.28 respectively. In the case of *D. melanogaster* chromosome 2L, the CDS/nonCDS threshold was increased until less than 20% of the nts in the chromosome where predicted as coding. In *C. elegans* chromosome I, the results shown were found when below 40% of nts were predicted as coding (also shown in results).

2.0.17 Unassembled sequences

Test sequences from unassembled or partially assembled reads or may not have a fraction of coding nucleotides representative of a parent species. For example, many datasets consist of short DNA sequences which are selected to contain single genes such as the EGASP dataset[38]. Due to the removal of vast amounts of intergenic DNA, the coding regions in this dataset may occupy significantly more than 3% of each sequence. In this case, the above strategy is not applicable and an estimate must be made when setting a CDS/nonCDS threshold. Extensive testing has shown strong patterns regarding the best values of this parameter which will be shown in the results section.

2.0.18 Adjustable threshold

For the CDS regions of any given organism, there is a mean length. Deviations from this mean become increasingly rare as the distance from the mean increases. A potential way to reduce the prediction of false positive CDS regions is to increase the threshold for potential CDS regions which a longer or shorter than the expected mean CDS length. As the mean length of CDS regions is fairly similar between closely related organisms, an estimate of the expected mean can often be made for an un-annotated genome.

When implemented however, adjusting the CDS/nonCDS threshold in this way had a large and definite negative impact on accuracy with over 0.1 drop in F1 score on the G3PO dataset. Further analysis was not conducted on this method. One possible reason for this result is that the initial predictions which were recorded before the CDS prediction step were innately accurate enough to limit the occurrence of overly long or short CDS regions. In other words, very long and very short potential CDS regions were not a source of false positives to begin with. Adding an additional barrier to these long or short regions merely resulted in the loss of true positives instead of

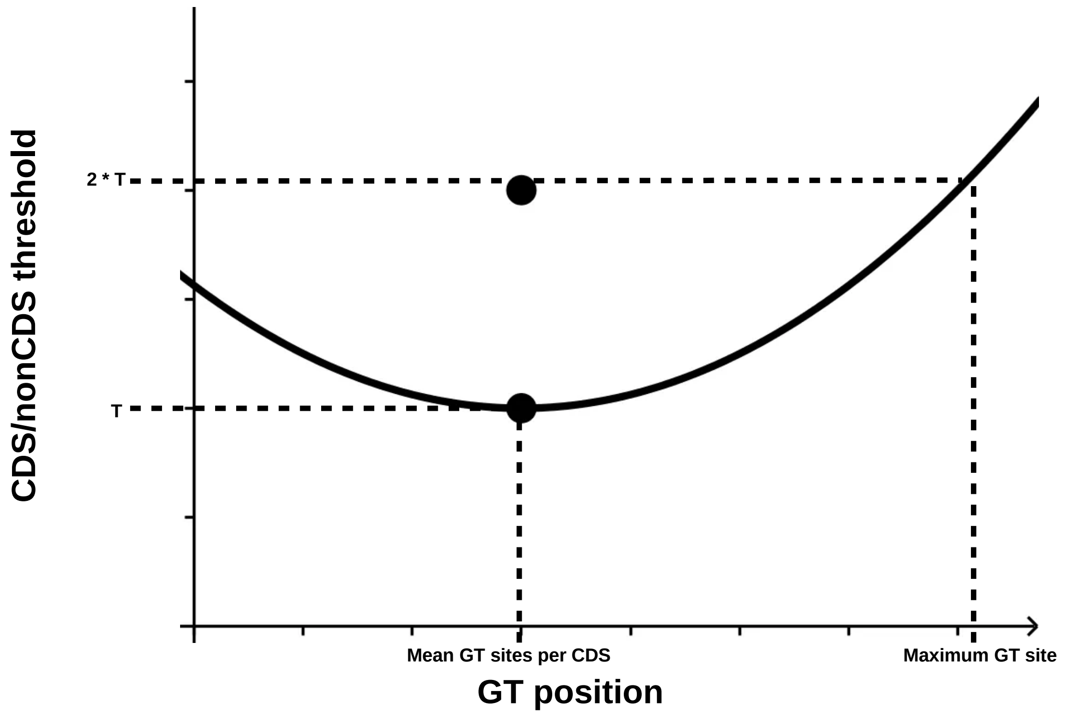


Figure 2.5: A potential method of restricting prediction of overly large or small CDS regions was to increase the voting threshold required for potential CDS regions which differed in length from a predicted mean value. This method was found to reduce accuracy significantly.

false positives.

2.0.19 Overlapping potential CDS regions

Start and stop codon motifs as well as splice site motifs such as YAG and GT are very likely to occur by chance in both intergenic and intronic regions of DNA. When generating a list of all possible CDS regions which are defined by these motifs, a large number of overlapping potential CDS regions are found. In nature however, true CDS regions which overlap are virtually non-existent. Several strategies for handling these overlapping regions have been explored with varying degrees of success. Three methods will be outlined in this section with data presented in the results section. These methods are referred to as *overlapping*, *best CDS score* and *non-overlapping.*

The most basic strategy and the one which was first implemented was to rely completely on the CDS/nonCDS voting threshold. In this method, any and all potential

CDS regions which encompassed a sufficient number of nts with positive class predictions was considered a true CDS region. If the initial predictions generated by the NN or RF were accurate enough, then there should theoretically be a proper CDS/nonCDS voting threshold which would separate overlapping CDS regions in such a way as to eliminate improper overlaps. This method performed well as has essentially outperformed other, more complex methods. This is called the *ovelapping* method.

The second method is to look at each family of overlapping potential CDS regions and choose only the only which has the highest CDS/nonCDS ratio. While seemingly logical, this method did not perform well. This is called the *best CDS score* method.

A third method is to set a minimum intron length and then reduce the search space for potential CDS regions by disallowing potential CDS regions which lie downstream from a previous potential CDS region unless the distance between the previous and current potential CDS regions is greater than the minimum intron length. This is called the *non-overlapping* method. A comparison of different methods of handling overlapping potential CDS regions is shown in the results section.

2.0.20 Alternate threshold definition

An alternate threshold definition was implemented for the latest RF based results. In the previous method, the CDS/nonCDS threshold at which a potential CDS was predicted as a true CDS was simply a ratio of the number of predicted coding nts inside a potential CDS versus the number of predicted non-coding nts. The alternate definition relates the fraction of positive predictions *inside* a potential CDS to the fraction of positive predictions in the *entire sample*:

$$\frac{Positive\ fraction\ (potential\ CDS)}{Positive\ fraction\ (entire\ sequence)} > Threshold$$

When dealing with whole chromosomes, the positive fraction of the entire sequence is the total number of positive nt predictions divided by the total length of the chromosome. In situations where small samples of DNA are being studied such as the G3PO dataset, the positive fraction of the entire sequence is the number of positive predictions divided by the total length of the sample sequence. This modification of the voting threshold was found to give a mean increase in F1 score of approximately 0.1 on the G3PO test dataset. This modification has only been applied to the RF based method.

2.0.21 Training Dataset Preparation

Training dataset formats for both NN and RF methods are identical. Each training dataset is composed of an $N_{fs} * N_s$ array where N_{fs} is the number of feature sensors employed and N_s is the number of nucleotide samples. Every nt in the training set is assigned a label of '1' if it belongs to a known CDS region and '0' if it does not. Nts which belong to CDS regions which are only sometimes part of a completed mRNA due to splicing are considered coding for both training and testing purposes.

Fully annotated chromosomes from the following model species were used to construct training datasets: textitDrosophila melanogaster (assembly BDGP6.32) Chromosome 2L, *Homo sapiens* (assembly GRCh38) chromosome 21, *Mus musculus* (assembly GRCm39) chromosome 19, *Caenorhabditis elegans* (assembly WBcel235) chromosome I, *Arabidopsis thaliana* (assembly TAIR10.54) chromosome 1 and *Saccharomyces cerevisae* (assembly R64-1-1.54) chromosome IV. All FASTA and GFF files used in datasets are available at https://ftp.ensembl.org/pub/release-103/fasta/ and http://ftp.ensembl.org/pub/release-103/gff3/.

Fully annotated chromosomes specify the locations and boundaries of all known CDS regions. The coding or non-coding status of every nt can therefore be determined and the chromosome is referred to as a 'labeled' chromosome. Each training dataset is generated by randomly selecting N_s labeled nucleotides from a single labeled chromosome. First, all CDS labeled (assigned 1) nucleotides are grouped and

N_1 of these are randomly selected without replacement. Second, all nonCDS labeled (assigned 0) nucleotides are grouped, and N_0 of these are randomly selected without replacement. No distinction between intergenic or intron regions are made for nonCDS labels. Note that $N_s = N_1 + N_0$.

Training on imbalanced sets of coding versus non-coding samples was observed to increase sensitivity at the expense of specificity (or vice versa), but these trade-offs were deemed a net loss in general. Therefore, in this work $N_1 = N_0 = N_s/2$. If the situation ever arises where a researcher specifically needs to increase Sn or Sp, the option to skew training data is effective.

In this work, each NN training set has ten million nts, or $N_s = 10^6$. Each set is further divided into fifths to generate five balanced training-validation sets. Each model is therefore trained on $8,000,000$ nucleotides with $2,000,000$ nucleotides used as validation. Data from NNs trained on 1 million nts is also included in some cases to serve as evidence that more training data most likely offers little advantage.

2.0.22 Model Organism Test Datasets

Each of the fully annotated chromosomes specified above serves as a test dataset. NNs or RFs which are trained on samples from one organism are used to predict all nts in each chromosome of the non-training organisms. Zero overlap between training and test organisms is present in these results.

2.0.23 G3PO Test Dataset

The use of 'standard' datasets against which gene prediction methods can be tested is well established. Unfortunately, the number of these datasets in existence makes them far from standard. In addition many well known datasets suffer from a lack of organism diversity. This becomes apparent when methods which perform well on simple datasets immediately lose accuracy when presented with more complex test sets. An common occurrence which has been noted by many authors[47, 37].

A highly diverse and well researched dataset known as the G3PO benchmark dataset was constructed in 2020[37]. It is composed of 1793 experimentally verified genes from 147 non-model eukaryotic organisms. The range of species is very broad and includes members of every eukaryotic kingdom. Accurate annotation was extensively researched by the authors. Each sequence in the dataset represents a single gene that includes all exons and introns, augmented by short segments of flanking DNA.

The G3PO dataset was constructed to serve as a bechmark test set to evaluate gene prediction methods on DNA from non-model organisms. The original paper compared the performance of five well known *ab initio* gene prediction methods: Genscan [66], GlimmerHMM [67], GeneID [68], Snap [69] and Augustus [10].

Mean nt level sensitivities(Sn) and precisions(PPV) across all species in the G3PO dataset were reported as: Genscan (Sn:0.50, PPV:0.57), GlimmerHMM (Sn:0.74, PPV:0.43), GeneID (Sn:0.38, PPV:0.52), Snap (Sn:0.38, PPV:0.45) and AUGUSTUS (Sn:0.51, PPV:0.58). Each method was run with default settings using organism specific models that are phylogenetically similar to each gene. These results are clear indication that the field of gene prediction has much room for improvement.

2.0.24 Evaluation of Performance: Sensitivity and Specificity

For the purposes of gene prediction, all nucleotides belong to one of two classes. The positive class includes nts which are inside a region of DNA which codes for protein. These regions are referred to as CDS regions. Some genes can be spliced in alternative patterns. Nucleotides within CDS regions which are only translated some of the time are still considered coding. Nucleotides which belong to regions of DNA which are never translated into protein are members of the negative class.

In evaluation of performance, the following definitions are therefore relevant. A true positive (TP) is a nucleotide which is known to belong to a CDS region and which is predicted to be in the positive class by the classifier. A fale positive (FP) is a nucleotide which does not belong to a CDS region and which is predicted to be in the positive class by the classifier. A true negative (TN) is a nucleotide which does not belong to a CDS region and which is predicted to be in the negative class by the classifier. A false negative (FN) is a nucleotide which does belong to a CDS region and which is predicted to be in the negative class by the classifier.

The technical definition of the term *accuracy* is the number of samples which are correctly classified versus the total number of samples. For samples which belong to only two classes:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{2.11}$$

Accuracy is not generally considered a useful metric in gene prediction due to the highly unbalanced ratio of coding to non-coding nts in most genomes. In unbalanced datasets such as these, very high accuracy can often be obtained trivially. For example, a method which simply predicts all nts as non-coding would achieve an accuracy of over 96% on human chromosome 21 which is composed of less than 3% coding nts. For this reason, other metrics are considered much more relevant.

The most common metrics for evaluating gene prediction performance are *sensitivity* (Sn) and *specificity* (Sp):

$$Sn = \frac{TP}{TP + FN} \tag{2.12}$$

$$Sp = \frac{TN}{TN + FP} \tag{2.13}$$

The arithmetic mean of Sn and Sp is known as *balanced accuracy* or BA. BA is not a commonly used metric of performance. In many situations however, a large number of results must be compared which have similar but slightly different values of both Sn and Sp. In these situations the BA is often a convenient metric (equation 2.10).

2.0.25 Evaluation of Performance: Precision and F1 score

The value of specificity as a measure of gene prediction accuracy has been debated. The designers of the EGASP gene prediction dataset argued that the large numbers of non-coding nucleotides present in most DNA made specificity a less relevant measure due to the large number of true negative classification results [47]. They argue that *precision* (PPV) is a more useful metric:

$$PPV = \frac{TP}{TP + FP} \tag{2.14}$$

In some cases, the mean between sensitivity and precision is a useful comparison tool. We refer to this metric as MSP:

$$MSP = \frac{Sn + PPV}{2} \tag{2.15}$$

The F1 score has been proposed and used by some authors as a single value measurement of gene prediction performance. The F1 score is defined as the harmonic mean of sensitivity(sometimes referred to as precision) and precision:

$$F1 = 2 * \frac{Sn * PPV}{Sn + PPV} \tag{2.16}$$

$$F1 = \frac{TP}{TP + \frac{1}{2}(FP + FN)} \tag{2.17}$$

However, the F1 score has been criticized as a metric due to its known dependency on the class balance of test data. Specifically, a test dataset which has a high proportion of negative class samples produces significant downward pressure on the F1 score[70]. The assessment of accuracy in gene prediction methods is highly dependent on the test dataset. It is common for methods to report their Sn and Sp values on small sets of curated DNA sequences. These datasets eliminate a large amount to non-coding intergenic DNA. This re-balancing of test data in the direction of positive samples is exactly what the authors of [70] suggested would cause an increase in F1 scores. For this reason, the F1 score, like other single number metrics of accuracy, is considered non-ideal.

CHAPTER 3: Results

3.0.1 Neural Network Results

One of the original goals of the NN based method was to achieve high *cross species* accuracy. A common flaw in gene prediction methods is that they learn patterns which are species specific. While this may be an advantage if an organism being studied has a closely related model organism which can provide training data, often the organism being studied is not closely related to any organisms with well annotated genomes. The ability to make accurate predictions when trained on species that are not closely related to the test organism is what we refer to as *cross species* accuracy.

The results in this section reflect this goal and in each case both test species and training species are indicated in order to ascertain cross species results. An extreme case is explored when NNs trained on DNA from single organisms are applied to ever gene in the G3PO dataset. Despite not selecting phylogenetically close training organisms, high accuracy was achieved on this dataset.

3.0.2 Neural Network: Training Results

Initial training consisted of a standard five fold validation process. Each training set was divided into five equal parts. Five neural networks were then trained on subsets of the training set which consisted of 4/5 of the total training set with the remaining 1/5 held in reserve for validation. It was found that the loss function values on the validation sets quickly dropped to a low value then leveled off. When comparing the validation set accuracies from all models after training, the mean was found to be 0.814 with a standard deviation of 0.016. This low SD implies that the choice of sample nts used to train the models does not significantly affect their final state.

The five fold validation process results in a total of 20 trained neural networks available for evaluation. All 20 of these models are evaluated on the test sets.

Training sets with greater than 1 million samples were found to give very little gain in accuracy. To demonstrate this point, results from NNs trained on both 800 thousand and 8 million samples are compared in this section. These numbers derive from the fact that only 80% of nts selected in a training set are every used at one time to train a given NN due to five fold cross validation (20% is always removed for validation. The mean increase in BA from 8 million training samples as opposed to 800 thousand samples was only 0.01. According to a paired t-test this increase was not significant with $p > 0.1$.

Some experimentation was conducted which used NNs trained on an uneven ratio of positive and negative samples. Over many trials it was found that an even ratio, where the number of positive and negative training samples were equal, gave the best results. As expected, the number of true positive increased when training data was skewed towards more positive samples but this was virtually always accompanied by a more significnt increase in false positives. An example of this is show for NNs trained on *C. elegans* chromosome I. For this reason, the results presented here are only for models trained on an equal number of positive and negative samples.

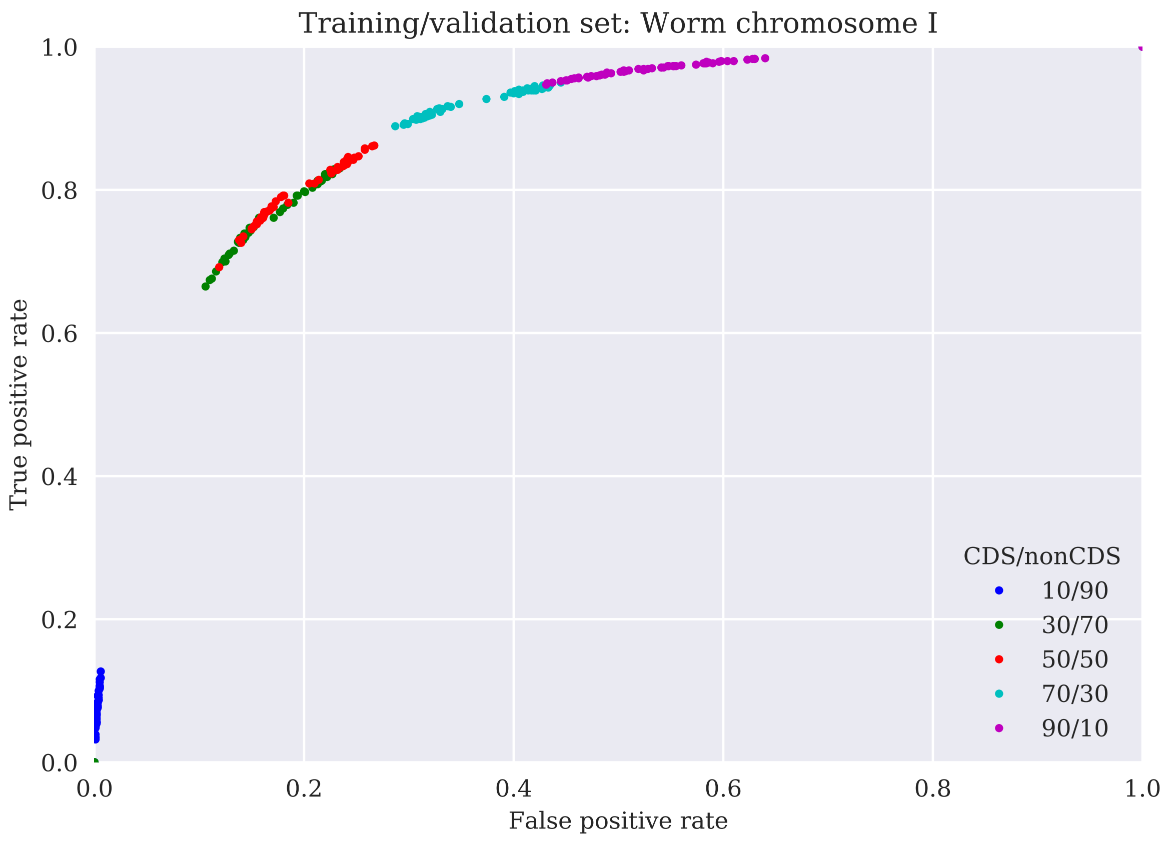


Figure 3.1: An example of the extremely predictable trade off between true and false positives as a function of the ratio of positive and negative samples from *C. elegans* chromosome I used to train a NN. A similar effect was found in NNs trained on all other species.

3.0.3 Neural Network: Model Organisms

These values each report the mean taken over five NNs, each trained on a different 80% subset of a 1 or 10 million nt training set. The standard deviations in Sn and Sp are low, almost always below 0.02. This fact was observed without exception and is evidence that the specific nts selected as training samples is relatively un-important.

Table 3.1: Accuracy results from a NN trained on **8 million** sample nts. These results were produced without the CDS prediction algorithm in order to compare them to results from a NN trained with more data. Mean values for all results: Sn = 0.72, Sp = 0.72, BA = 0.72.

	D. mel. (test)	*H. sap.* (test)	*M. mus.* (test)	*C. ele.* (test)
D. melanogaster (training)	Sn:0.78 Sp:0.84 BA:0.81	Sn:0.83 Sp:0.53 BA:0.68	Sn:0.75 Sp:0.65 BA:0.7	Sn:0.47 Sp:0.9 BA:0.68
H. sapiens (training)	Sn:0.63 Sp:0.78 BA:0.7	Sn:0.94 Sp:0.61 BA:0.77	Sn:0.67 Sp:0.78 BA:0.73	Sn:0.73 Sp:0.65 BA:0.69
M. musculus (training)	Sn:0.85 Sp:0.64 BA:0.74	Sn:0.45 Sp:0.95 BA:0.7	Sn:0.75 Sp:0.82 BA:0.78	Sn:0.65 Sp:0.75 BA:0.7
C. elegans (training)	Sn:0.74 Sp:0.72 BA:0.73	Sn:0.84 Sp:0.46 BA:0.65	Sn:0.65 Sp:0.66 BA:0.66	Sn:0.84 Sp:0.75 BA:0.8

After obtaining initial nucleotide level predictions from the NN, CDS regions are predicted by increasing the CDS/nonCDS threshold until the predicted coding fraction is below an estimate of the known coding fraction of a similar species. The coding fraction is the fraction of nucleotides which belong to coding regions inside a sequence. For the results in Table 1 the target coding fractions are: *D. melanogaster*: 0.2, *H. sapiens*: 0.1, *M. musculus*: 0.1 and *C. elegans*: 0.4. Table 3 lists Sn and Sp when trained and tested using all training/testing combinations of the four species.

The CDS algorithm gives a small mean gain in BA beyond the raw NN predictions. However, according to a paired t-test, this gain is not significant on average with

Table 3.2: Accuracy results from a NN trained on **800 thousand** sample nts. These results were produced without the CDS prediction algorithm in order to compare them to results from a NN trained with more data. Mean values for all results: Sn = 0.77, Sp = 0.65, BA = 0.71.

	D. mel. (test)	*H. sap.* (test)	*M. mus.* (test)	*C. ele.* (test)
D. melanogaster (training)	Sn:0.9 Sp:0.69 BA:0.8	Sn:0.85 Sp:0.51 BA:0.68	Sn:0.79 Sp:0.61 BA:0.7	Sn:0.52 Sp:0.88 BA:0.7
H. sapiens (training)	Sn:0.56 Sp:0.85 BA:0.7	Sn:0.93 Sp:0.56 BA:0.75	Sn:0.55 Sp:0.88 BA:0.72	Sn:0.58 Sp:0.78 BA:0.68
M. musculus (training)	Sn:0.96 Sp:0.39 BA:0.68	Sn:0.78 Sp:0.75 BA:0.76	Sn:0.93 Sp:0.51 BA:0.72	Sn:0.9 Sp:0.48 BA:0.69
C. elegans (training)	Sn:0.78 Sp:0.68 BA:0.73	Sn:0.86 Sp:0.49 BA:0.68	Sn:0.69 Sp:0.62 BA:0.66	Sn:0.8 Sp:0.77 BA:0.78

$p > 0.1$. It was observed that the majority of cases where BA decreased were instances where *M. musculus* was used as either training or test organism. The potential gain in accuracy from the CDS prediction algorithm is only part of its purpose as this step also results in biologically sound predictions which are properly bounded by start, stop and splice site motifs.

A high CDS/nonCDS threshold will cause Sn to decrease while a low threshold will cause Sn to increase. Sp is affected in the reverse direction. For any CDS/nonCDS threshold the overall gain outweighs the overall loss. In many instances, an increase in Sn is accompanied by a decrease in Sp. In these situations the balanced accuracy (BA) is a useful way of determining if the net overall change is a gain or loss.

Table 3.3: Accuracy results from a NN trained on **8 million** samples **with** CDS algorithm processing. CDS/nonCDS thresholds at which results were measured are given. Mean values for all results: Sn = 0.73, Sp = 0.74, BA = 0.74.

	D. mel. (test)	*H. sap.* (test)	*M. mus.* (test)	*C. ele.* (test)
D. melanogaster (training)	Sn:0.82 Sp:0.87 BA:0.84 Th: 10	Sn:0.78 Sp:0.71 BA:0.74 Th: 18	Sn:0.61 Sp:0.81 BA:0.71 Th: 29	Sn:0.78 Sp:0.73 BA:0.76 Th: 1
H. sapiens (training)	Sn:0.66 Sp:0.83 BA:0.74 Th: 4	Sn:0.77 Sp:0.75 BA:0.76 Th: 10	Sn:0.58 Sp:0.88 BA:0.73 Th: 4	Sn:0.76 Sp:0.68 BA:0.72 Th: 2
M. musculus (training)	Sn:0.88 Sp:0.56 BA:0.72 Th: 29	Sn:0.61 Sp:0.92 BA:0.76 Th: 24	Sn:0.93 Sp:0.5 BA:0.72 Th: 29	Sn:0.76 Sp:0.67 BA:0.72 Th: 23
C. elegans (training)	Sn:0.63 Sp:0.84 BA:0.74 Th: 21	Sn:0.73 Sp:0.62 BA:0.68 Th: 29	Sn:0.46 Sp:0.81 BA:0.64 Th: 25	Sn:0.89 Sp:0.71 BA:0.8 Th: 5

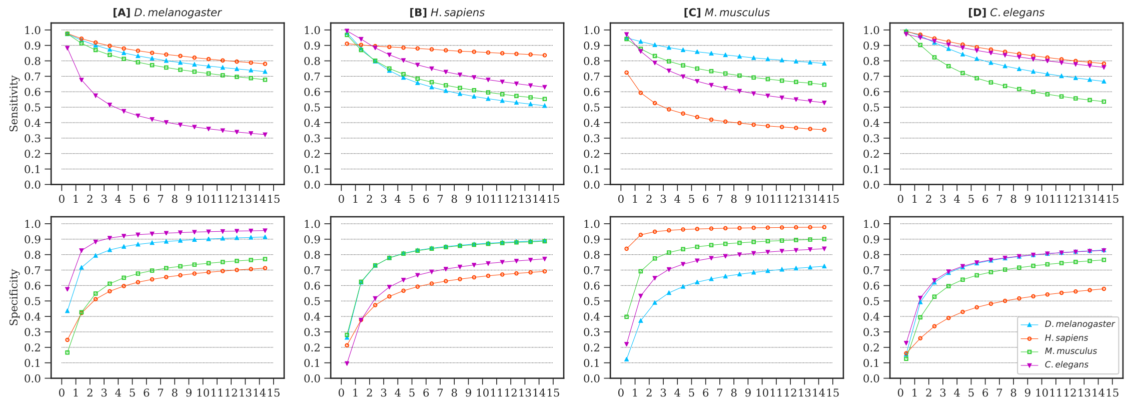
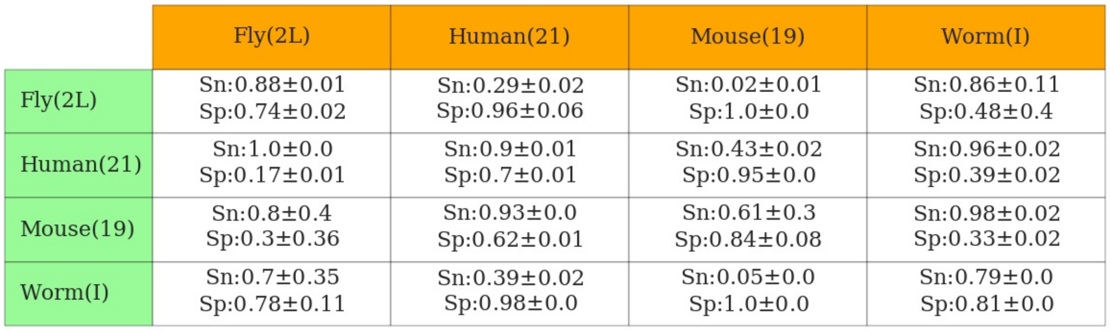


Figure 3.2: Sn and Sp curves as functions of CDS/nonCDS threshold. Each column represents a single test species. For each test species, predictions from NNs trained on different species DNA were compared indicated by different colors.

3.0.4 Neural Network: Alternative architectures

The method presented here is based on a dense neural network. However, many attempts were made to construct a method based on recurrent neural network (RNN) architecture. The networks used LSTM nodes in place of standard fully connected neurons. The results from these experiments showed promise, but had an unpredictable tendency to suddenly become an all or nothing classifier. Some trained RNN models would simply predict all nts were coding or all nts were noncoding. This resulted in extreme values for sensitivity or specificity. This phenomenon was not predictable beforehand. Several RNNs trained on the same data would not all become trapped in this way. For this reason it was decided to direct attention away the RNN based models.



	Fly(2L)	Human(21)	Mouse(19)	Worm(I)
Fly(2L)	Sn:0.88±0.01 Sp:0.74±0.02	Sn:0.29±0.02 Sp:0.96±0.06	Sn:0.02±0.01 Sp:1.0±0.0	Sn:0.86±0.11 Sp:0.48±0.4
Human(21)	Sn:1.0±0.0 Sp:0.17±0.01	Sn:0.9±0.01 Sp:0.7±0.01	Sn:0.43±0.02 Sp:0.95±0.0	Sn:0.96±0.02 Sp:0.39±0.02
Mouse(19)	Sn:0.8±0.4 Sp:0.3±0.36	Sn:0.93±0.0 Sp:0.62±0.01	Sn:0.61±0.3 Sp:0.84±0.08	Sn:0.98±0.02 Sp:0.33±0.02
Worm(I)	Sn:0.7±0.35 Sp:0.78±0.11	Sn:0.39±0.02 Sp:0.98±0.0	Sn:0.05±0.0 Sp:1.0±0.0	Sn:0.79±0.0 Sp:0.81±0.0

Figure 3.3: Results from a small, two layer RNN. While some results are good, there are clearly some intances where the RNN became trapped in an over or under predicting mode. This can be seen in some of the results which have extremely low Sn or Sp values.

3.0.5 Neural Network: G3PO Test Dataset

We applied the same 20 distinct trained NNs from above to every gene in the G3PO dataset. For each set of five NNs which were trained on the same species (see Neural Network: Training Results above) we predicted coding or non-coding status of every nt in each sample gene. Results are separated by training species.

An optimal CDS/nonCDS threshold could be not be set for each gene using the predicted coding fraction method. This inherently puts our method at a disadvantage. For reasons given above, the method being developed here relies on an estimate of the coding fraction in sample DNA. In practice, this estimate can be made when the sample DNA is a complete chromosome. In this case, CDS predictions are evaluated as the CDS/nonCDS threshold is varied from 1/3 to 15. At each iteration, the threshold is held fixed to observe the mean results at that threshold across the whole dataset.

As already stated, our method requires a complete chromosome to make estimates of an optimal CDS/nonCDS threshold by observing the predicted coding fractions. Yet even in a situation where no estimate can be made, there is a range of CDS/nonCDS thresholds for which Sn and Sp values are equal to or higher than those produced by many the other *ab initio* methods tested in [37].

At a fixed threshold of 0.7, NNs trained on *H. sapiens* and *M. musculus* were able to clearly outperform methods GeneID, GlimmerHMM and Snap while only trailing slightly behind methods AUGUSTUS and Genscan. This can be seen in the table of results presented in this section.

As shown in figure 3.4, across all genes, the mean Sn decreases and mean PPV increases as the CDS/nonCDS threshold increases. This figure shows a clear trend that high threshold values are not ideal on average. Low threshold values were consistently found to be, on average, superior in terms of Sn and PPV. To further examine this, distributions were generated which tracked the best threshold values in terms of F1 score and balanced accuracy for each gene in the dataset. It can clearly be seen that

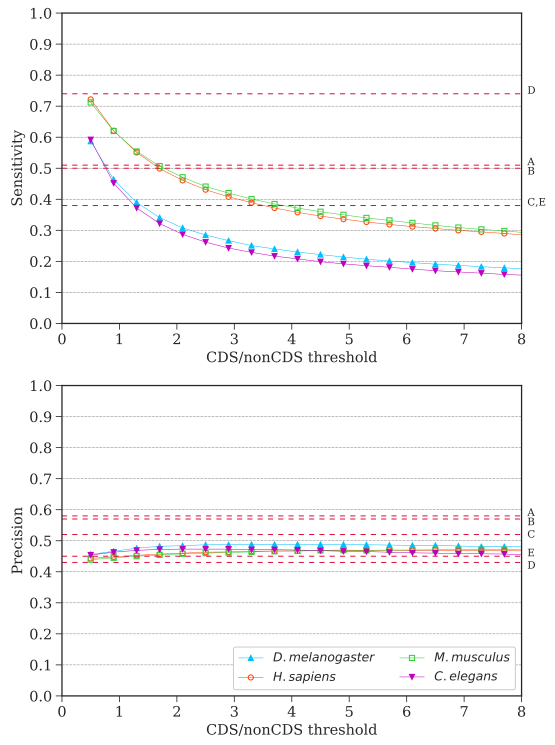


Figure 3.4: Mean G3PO accuracy metrics as a functions of fixed CDS/nonCDS thresholds (first type). Red dotted lines indicate mean values from competing methods: A) AUGUSTUS, B) Genscan, C) GeneID, D) GlimmerHMM, E) Snap.

high fixed thresholds are not ideal.

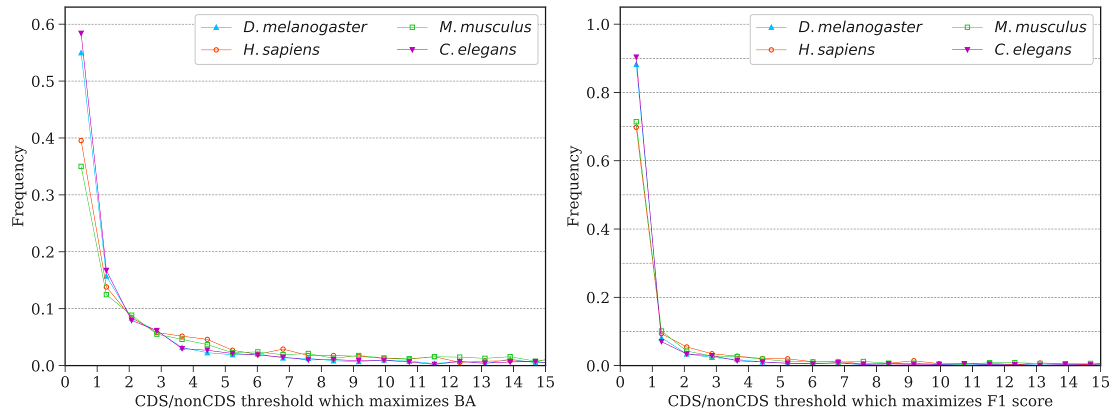


Figure 3.5: Distribution of CDS threshold values which maximize balanced accuracy and F1 score per gene.

When dealing with whole chromosomes, an estimate of the true CDS fraction allows the CDS/nonCDS threshold to be set with much greater confidence. As this was not the case when using a single, fixed threshold across all genes, the mean values of Sn, PPV, MSP and F1 score are below what could be expected when better estimates could be made on a per organism basis. To gain some insight into how well this method could perform on a broad range of species in situations where superior estimates could be made, we have included the mean maximum F1 scores and MSP values which are obtained on the G3PO dataset. These values were calculated by scanning the results of each gene individually and selecting the threshold which maximized either F1 score or MSP. The mean of these maximum scores across all genes gives an upper bound estimate which could be obtained with individual threshold settings. These numbers show that with correct threshold settings, this method has the potential to outperform other methods.

Table 3.4 is a summary of accuracy metrics over the G3PO dataset. Values for the five comparison methods are taken from[37]. Rows which are labeled by a species name followed by the word FIXED are results for a single fixed threshold value applied to every gene in the dataset. Rows labeled MSP or F1 show values when thresholds are selected on a per gene basis which maximise either the mean of Sn and PPV (MSP) or the F1 score. For these rows, the mean threshold which was chosen is given as well as the standard deviation of these threshold values. In all cases low values were observed to have the best mean performance.

With fixed thresholds, none of the NNs were able to defeat all five competing methods in terms of F1 score. However, models trained on *H. sapiens* or *M. musculus* DNA were both able to achieve F1 scores of 0.47 which is greater than the F1 scores of methods GeneID, GlimmerHMM and Snap. When allowed to vary threshold values in order to maximize F1 scores these models were able to achieve higher mean F1 scores than all five competing methods.

Table 3.4: Summary of Sn, PPV, MSP and F1 on the GP3O dataset for the five top performing methods and the sensor-NN method for three cases: when the CDS/nonCDS threshold is fixed at 0.7 or set to maximize MSP or F1 scores. The mean CDS/nonCDS threshold required to maximize the target metric and the standard deviation of these threshold values are also reported.

	Sn	PPV	MSP	F1	Th	SD
AUGUSTUS	0.51	0.58	0.55	0.52	NA	NA
Genscan	0.50	0.57	0.54	0.51	NA	NA
GeneID	0.38	0.52	0.45	0.40	NA	NA
GlimmerHMM	0.74	0.43	0.56	0.45	NA	NA
Snap	0.38	0.45	0.42	0.39	NA	NA
D. melanogaster MSP	0.69	0.50	0.59	0.47	1.75	4.88
D. melanogaster F1	0.67	0.54	0.60	0.51	3.41	5.73
D. melanogaster Fixed	0.60	0.45	0.53	0.43	0.7	0
H. sapiens MSP	0.84	0.48	0.66	0.53	2.11	5.34
H. sapiens F1	0.71	0.53	0.62	0.54	3.75	5.85
H. sapiens Fixed	0.74	0.44	0.59	0.47	0.7	0
M. musculus MSP	0.80	0.48	0.64	0.52	2.36	5.84
M. musculus F1	0.83	0.42	0.63	0.55	5.31	6.71
M. musculus Fixed	0.74	0.44	0.59	0.47	0.7	0
C. elegans MSP	0.71	0.50	0.61	0.47	1.48	4.47
C. elegans F1	0.66	0.54	0.60	0.49	2.93	5.36
C. elegans Fixed	0.60	0.46	0.53	0.41	0.7	0

3.0.6 Random Forest Results

The design goal of the RF based method was slightly different than that of the NN based method. Rather than attempting to achieve cross species accuracy, the goal was to obtain *universal* accuracy. A combination of several RF classifiers were combined. Each RF was trained on a different model species. The combined RFs were then tested against several whole chromosomes from each model organism. Chromosomes used for training were not used as test chromosomes.

An expanded set of model organisms were chosen to serve as training and test organisms for the RF based method in an attempt to make the method more universal. New organisms include *Arabidopsis thaliana* and *Saccharomyces cerevisae*. These introduce genomes from plant and fungal kingdoms which were not present in the NN results. The decision was made to remove *M. musculus* DNA from both training and test sets. It was found that despite the mouse being a model organism, its genome was not nearly as well annotated as the other model organisms being used.

3.0.7 Random Forest: Training Results

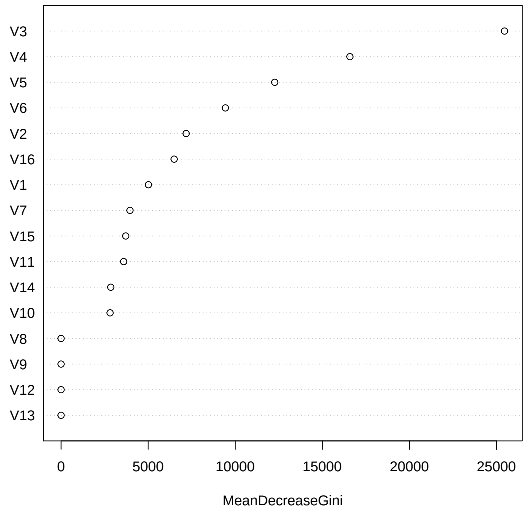
Five random forests were trained on DNA from different model species. These species and the chromosomes used to generate the training sets include: textit-Drosophila melanogaster (assembly BDGP6.32) Chromosome 2L, *Homo sapiens* (assembly GRCh38) chromosome 21, *Caenorhabditis elegans* (assembly WBcel235) chromosome I, *Arabidopsis thaliana* (assembly TAIR10.54) chromosome 1 and *Saccharomyces cerevisae* (assembly R64-1-1.54) chromosome IV. All FASTA and GFF files used in datasets are available at https://ftp.ensembl.org/pub/release-103/fasta/ and http://ftp.ensembl.org/pub/release-103/gff3/. As before, nts which lie inside a known CDS region according to the GFF files are labeled as coding regardless of whether or not the CDS belongs to a gene which optionally transcribes the CDS due to alternative splicing.

Each RF consisted of 2 thousand decision trees and was trained on 100 thousand randomly selected sample nts. A 50/50 split between coding and non-coding nts was chosen to train each RF. Training of each tree used the maximum branch purity method which minimized Gini index at each branch. 100 thousand samples was deemed sufficient as gains in accuracy beyond this point were virtually zero. A table of values is shown below which compares several accuracy metrics from RFs trained on 25, 50 and 100 thousand samples. The results are convincing evidence that using more training data is of minimal value.

The 16 sensors used in the NN method above were used to train several RF classifiers. The variable importance calculation results indicated that some sensors were far more useful to the RF when fitting the training data than others. The k6 sensor, which indicates the preference of 6-letter k-mer word usage inside each nucleotide window was shown to be the most important determinant of RF accuracy by far. All eight of the least important sensors were those which indicated the proximity to start, stop or splice site motifs. Based on these observations some changes were made to the range of sensors used in the RF based method.

The eight least important sensors as indicated by mean Gini index decrease were removed from the RF model. In addition, the GC sensor was removed and replaced by four sensors which each measured the content of a single base A, C, G or T. The DFT L3 sensor which measures periodicity three signal was replaced by four independent auto-correlation sensors. These changes were made due to observations of Fickett in his gene prediction method[50]. This set of sensors is referred to as the '14 sensor' model, while the previous set of sensors used in the NN results above is referred to as the '16 sensor' model.

As a side note, experiments were performed which used NNs trained on the 14 sensor model but a consistent slight decrease in accuracy was observed relative to the 16 sensor model and further experimentation with NNs using this combination was



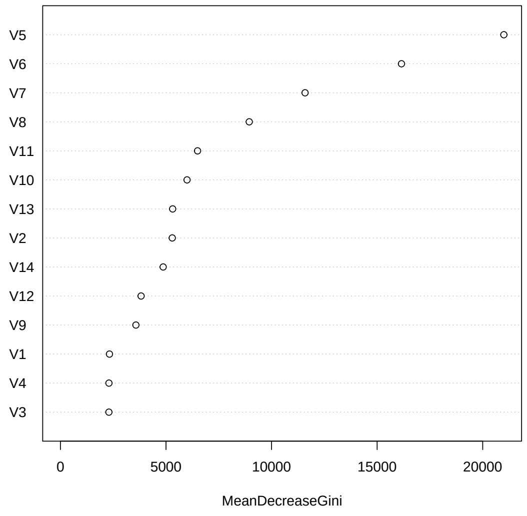


Figure 3.6: **LEFT:** 16 sensor feature importance in order of mean Gini index decrease. V1) DFT L3 sensor, V2) GC content sensor, V3) k6 sensor, V4) k5 sensor, V5) k4 sensor, V6) k3 sensor, V7) k2 sensor, V8) Donor splice site sensor right, V9) Acceptor splice site sensor right, V10) Start codon sensor right, V11) Stop codon sensor right, V12) Donor splice site sensor left, V13) Acceptor splice site sensor left, V14) Start codon sensor left, V15) Stop codon sensor left, V16) Entropy sensor. **RIGHT:** 14 sensor feature importance in order of mean Gini index decrease. V1) A content sensor, V2) C content sensor, V3) G content sensor, V4) T content sensor, V5) k6 sensor, V6) k5 sensor, V7) k4 sensor, V8) k3 sensor, V9) k2 sensor, V10) Entropy sensor, V11) A auto-correlation sensor, V12) C auto-correlation sensor, V13) G auto-correlation sensor, V14) T auto-correlation sensor

not pursued. This is not a strange result as the methods by which RFs use to classify data are dramatically different than NNs. A RF trained by Gini index reduction may have no use for certain features while a NNs may be capable of finding some hidden correlation with other features.

3.0.8 Random Forest: Model Organisms

Sensitivity (Sn), specificity (Sp), balanced accuracy (BA) and F1 scores are given in the table below. Four chromosomes from each model species are shown. The predicted CDS fraction (CDSF) is also shown to as this is a primary source of feedback which can be used to select a proper threshold. As threshold values are increased, all of these numbers are affected. From the more detailed analysis which follows, it will be seen that the best values for the threshold always fall in a zone which makes predictions such that the CDSF is close to the known fraction of coding DNA. This table gives an overview of results at specific values of the threshold in order to give an overview of results.

Table 3.5: RF results over four chromosomes from each model species.

Chromosome	Sn	Sp	BA	F1	CDSF	Th
D. melanogaster 2R	0.75	0.88	0.82	0.5	0.18	3.3
D. melanogaster 3L	0.72	0.9	0.81	0.49	0.15	3.7
D. melanogaster 3R	0.73	0.9	0.82	0.52	0.16	3.5
D. melanogaster 4	0.69	0.87	0.78	0.45	0.18	4.6
H. sapiens 18	0.63	0.85	0.74	0.03	0.15	4.3
H. sapiens 19	0.6	0.85	0.72	0.13	0.16	3.1
H. sapiens 20	0.68	0.85	0.76	0.06	0.16	3.5
H. sapiens 22	0.77	0.83	0.8	0.06	0.17	3.7
C. elegans II	0.8	0.8	0.8	0.53	0.28	3.0
C. elegans III	0.8	0.85	0.82	0.58	0.24	3.5
C. elegans IV	0.78	0.84	0.81	0.53	0.24	3.5
C. elegans V	0.67	0.83	0.75	0.48	0.24	3.5
S. cerevisae VII	0.95	0.63	0.79	0.75	0.6	1.2
S. cerevisae XII	0.89	0.57	0.73	0.47	0.52	1.6
S. cerevisae XIII	0.88	0.71	0.8	0.73	0.5	1.7
S. cerevisae XV	0.89	0.59	0.74	0.51	0.51	1.7
A. thaliana 2	0.76	0.8	0.78	0.48	0.27	3.2
A. thaliana 3	0.73	0.82	0.77	0.51	0.26	3.2
A. thaliana 4	0.75	0.8	0.78	0.5	0.27	3.1
A. thaliana 5	0.77	0.81	0.79	0.53	0.28	3.1

In the next section, a more detailed analysis of accuracy measures is given for each model species. An important observation is that maximum values for BA and F1 scores sometimes occur when predicted CDSF values are above or below the known coding fraction of the test species. In cases such as *H. sapiens* where the true coding fraction of the chromosomes lie between 0.01 and 0.06, the maximal values of BA an F1 score occur when predicted CDSF is above the known fraction. This is due to the fact that DNA which has very a low fraction of coding nts present far more opportunities for false positives than it does for true positives. Increasing the threshold reduces both true and false positives. Is cases such as *S. cerevisae* , which have over 0.7 known coding fraction, the best results occur at thresholds which give under 0.5 or 0.6 predicted CDSF for the inverse reason.

3.0.9 Random Forest: *D. melanogaster*

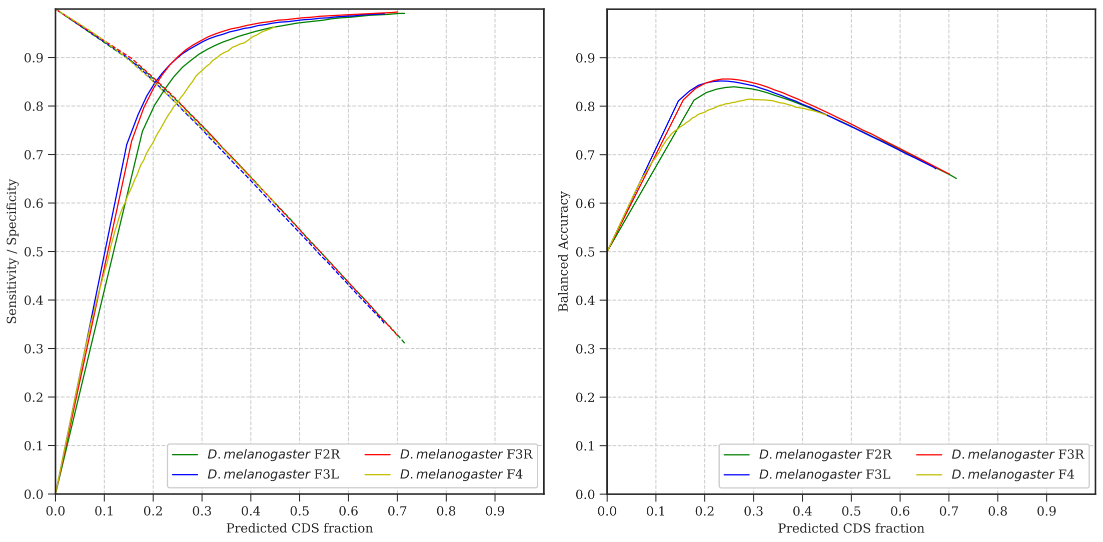


Figure 3.7: Sensitivity, Specificity and balanced accuracy as function of predicted CDS fraction for *D. melanogaster* chromosomes. The known CDS fraction of the *D. melanogaster* genome is 0.17[1].

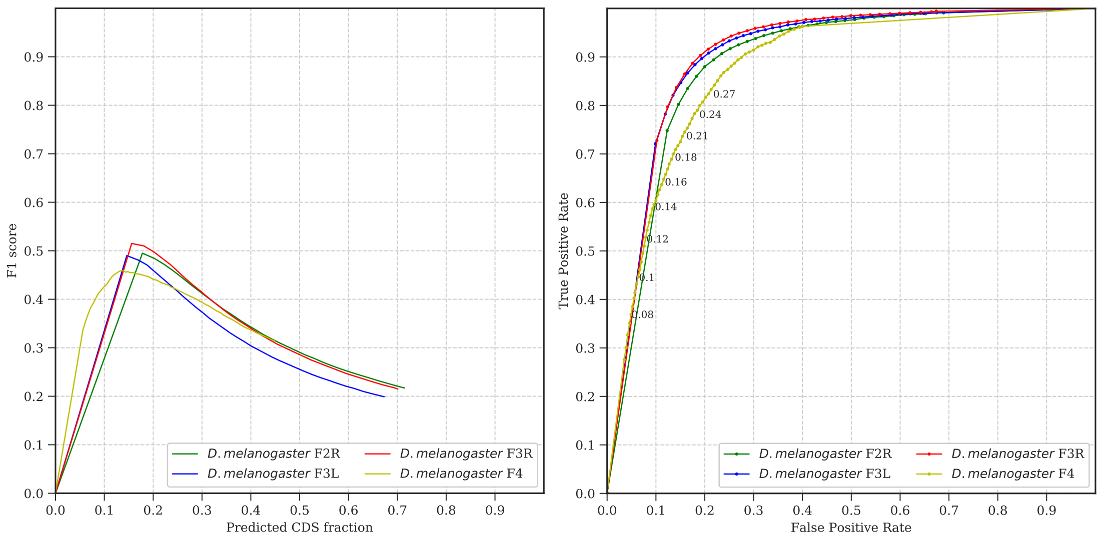


Figure 3.8: F1 score and ROC curve as function of predicted CDS fraction for *D. melanogaster* chromosomes. The known CDS fraction of the *D. melanogaster* genome is 0.17[1]. Shown on the ROC curve are some values of predicted CDSF which lie close to the known coding fraction.

Several alternative methods of dealing with overlapping potential CDS regions were explored (See methods section). A clear pattern was observed which indicated that allowing the model to predict overlapping CDS regions gave the best overall accuracy. The exact reason as to why this was observed is not clear but several possible reasons exist. A figure below compares the F1 score on a single *D. melanogaster* chromosome which demonstrates this result.

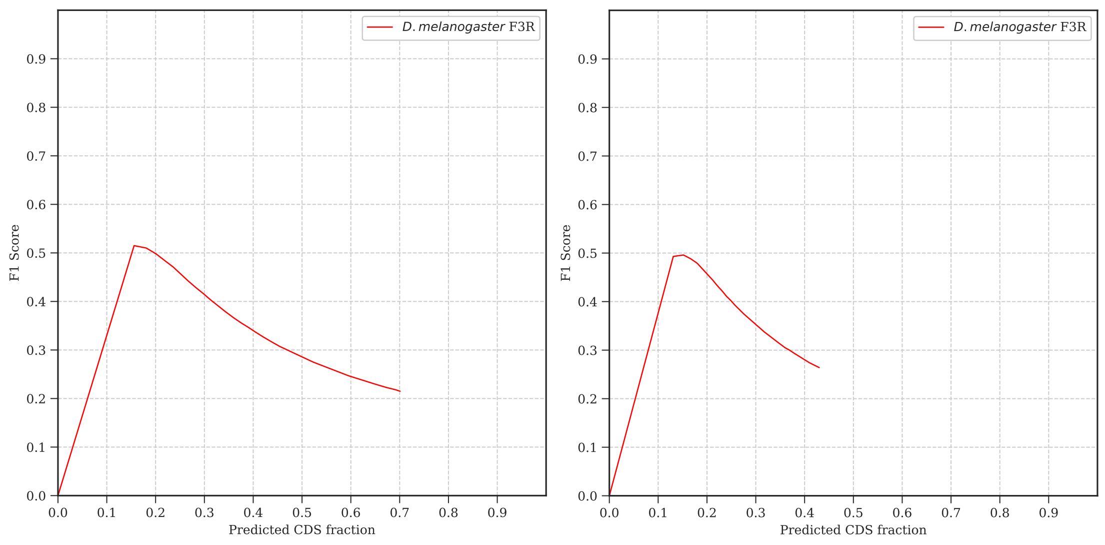


Figure 3.9: F1 score comparison between different methods of handling overlapping potential CDS regions on *D. melanogaster* chromosome F3R. The plot on the left is from a method which allows overlapping CDS regions to be predicted. The plot on the right does not allow overlapping CDS regions and forces a minimum distance between adjacent CDS regions.

One possible reason is that the distribution of positive predictions inside some potential CDS regions is not perfectly even. Is situations were small sections of a larger CDS region have false negative 'holes', multiple overlapping CDS regions will look better to the CDS prediction algorithm. If overlapping CDS regions are eliminated, the it is possible that multiple overlapping regions which look like true CDS regions will be replaced with a single large CDS regions which does not look like a true CDS region. Limited but inconclusive evidence for this reasoning was seen in the drop in true positives which accompanied all attempts to eliminate overlapping CDS regions. In all cases observed so far, allowing overlapping regions was found to give the best results.

3.0.10 Random Forest: *H. sapiens*

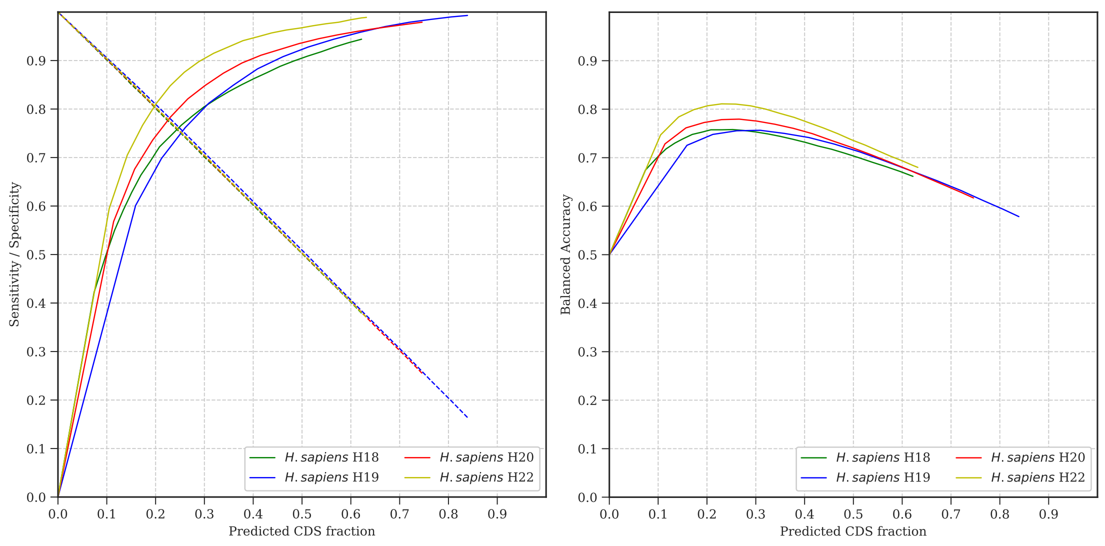


Figure 3.10: Sensitivity, Specificity and balanced accuracy as function of predicted CDS fraction for *H. sapiens* chromosomes. The known CDS fraction of the *H. sapiens* genome is 0.02[1].

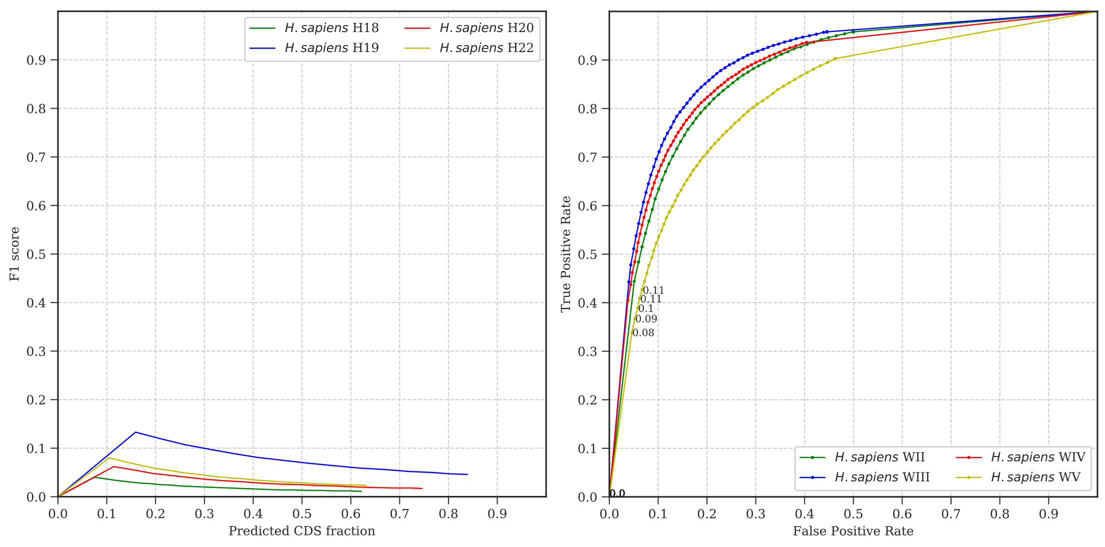


Figure 3.11: F1 score and ROC curve as function of predicted CDS fraction for *H. sapiens* chromosomes. The known CDS fraction of the *H. sapiens* genome is 0.02[1]. Shown on the ROC curve are some values of predicted CDSF which lie close to the known coding fraction.

3.0.11 Random Forest: *C. elegans*

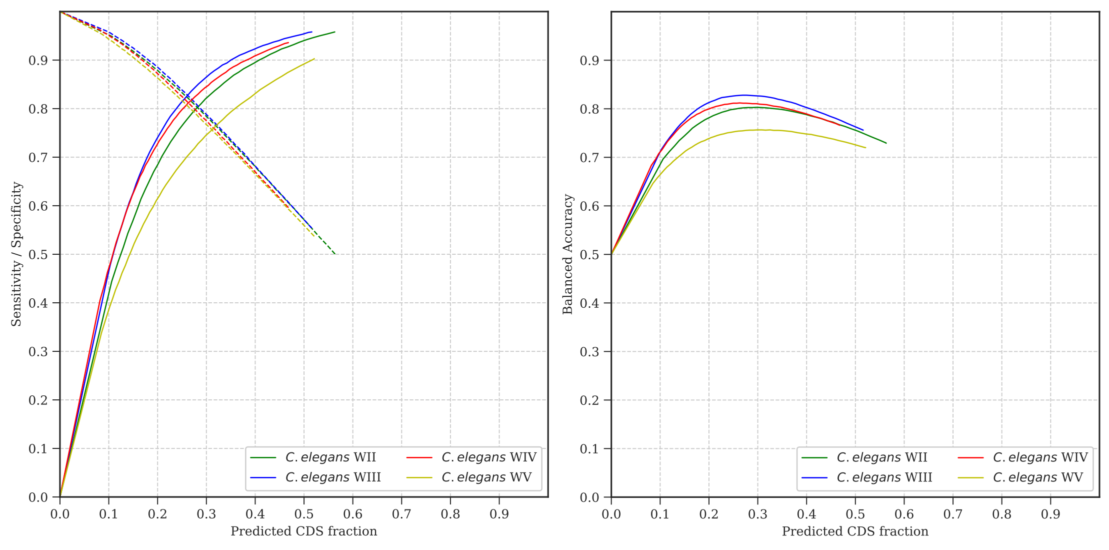


Figure 3.12: Sensitivity, Specificity and balanced accuracy as function of predicted CDS fraction for *C. elegans* chromosomes. The known CDS fraction of the *C. elegans* genome is 0.28[1].

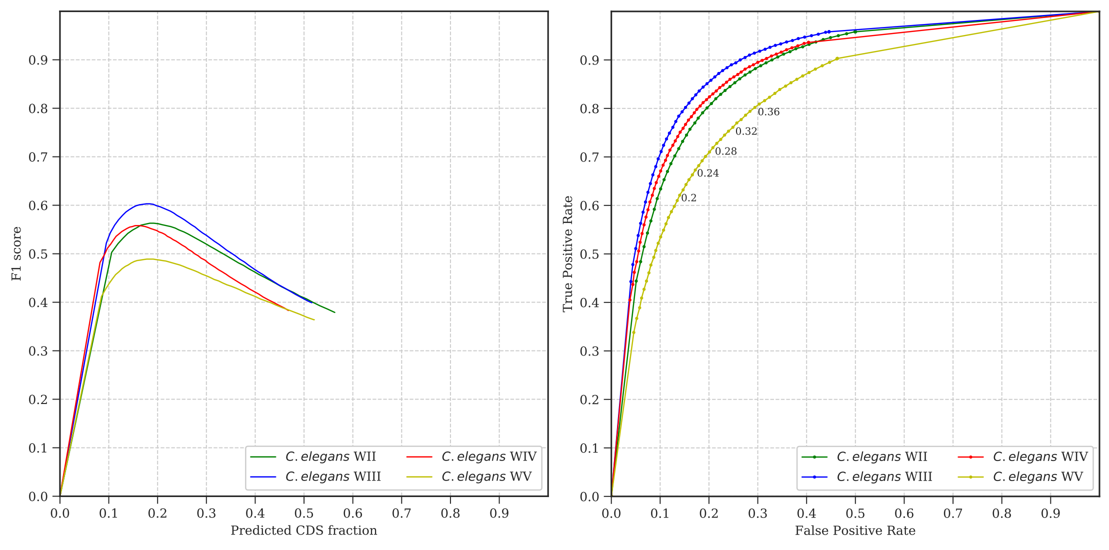


Figure 3.13: F1 score and ROC curve as function of predicted CDS fraction for *C. elegans* chromosomes. The known CDS fraction of the *C. elegans* genome is 0.28[1]. Shown on the ROC curve are some values of predicted CDSF which lie close to the known coding fraction.

3.0.12 Random Forest: *S. cerevisae*

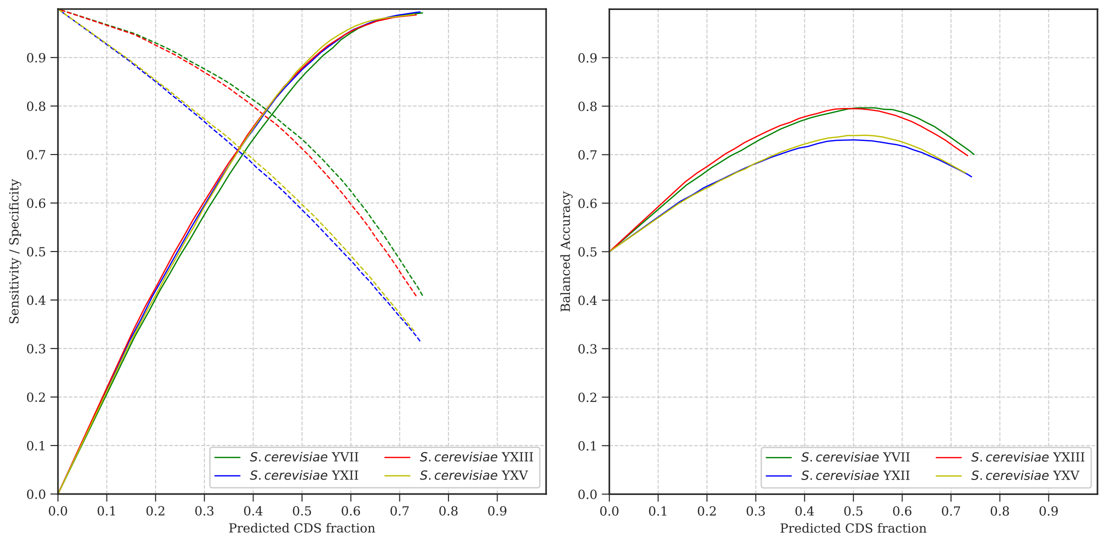


Figure 3.14: Sensitivity, Specificity and balanced accuracy as function of predicted CDS fraction for *S. cerevisae* chromosomes. The known CDS fraction of the *S. cerevisae* genome is 0.72[1].

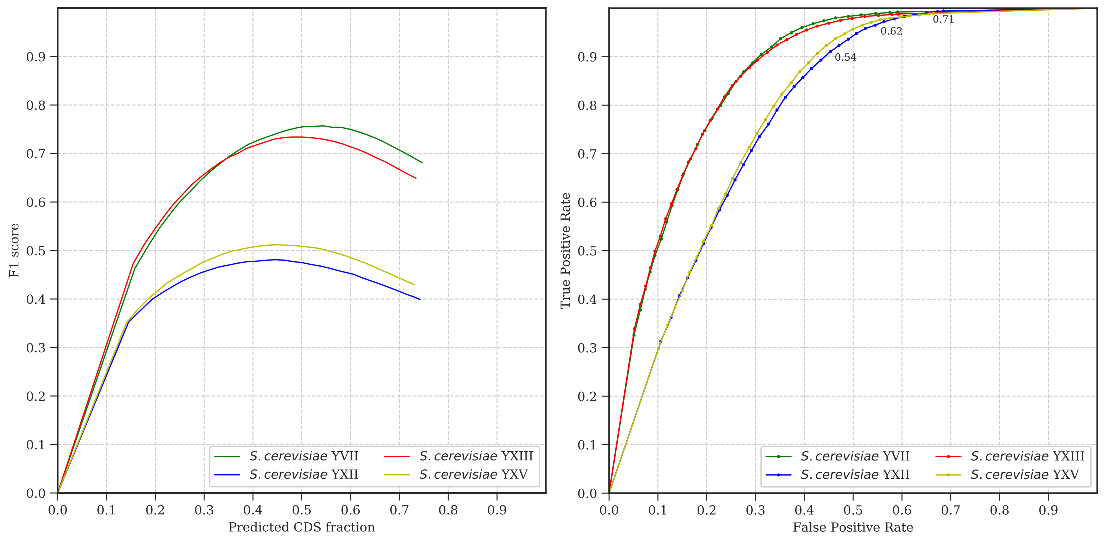


Figure 3.15: F1 score and ROC curve as function of predicted CDS fraction for *S. cerevisae* chromosomes. The known CDS fraction of the *S. cerevisae* genome is 0.72[1]. Shown on the ROC curve are some values of predicted CDSF which lie close to the known coding fraction.

3.0.13 Random Forest: *A. thaliana*

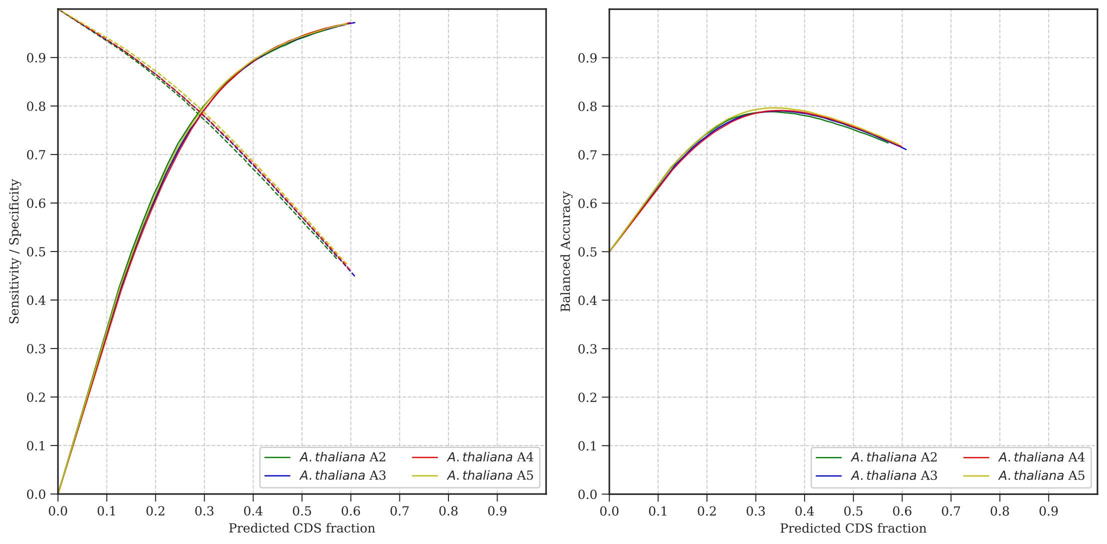


Figure 3.16: Sensitivity, Specificity and balanced accuracy as function of predicted CDS fraction for *A. thaliana* chromosomes. The known CDS fraction of the *A. thaliana* genome is 0.32[1].

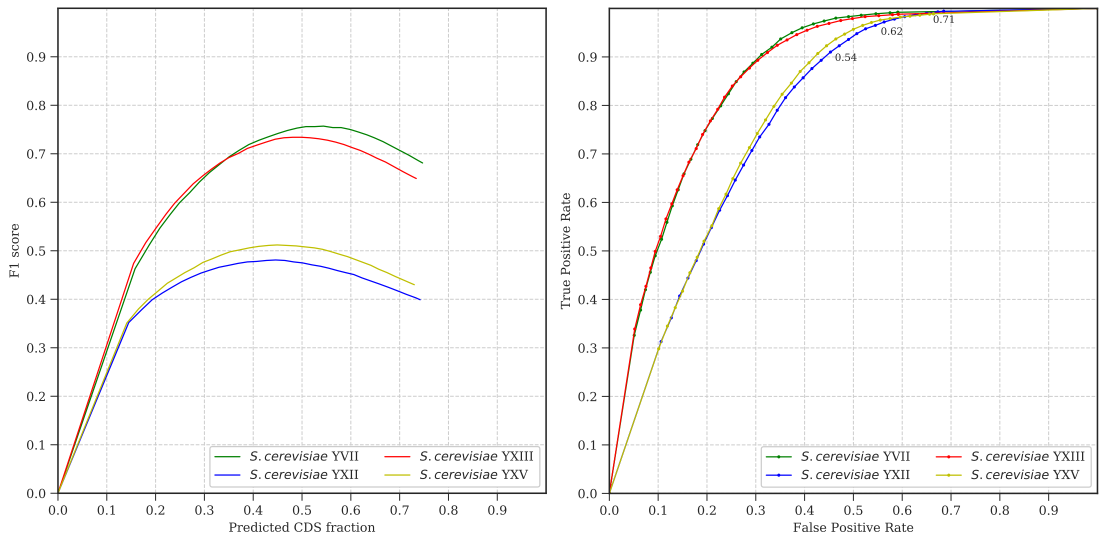


Figure 3.17: F1 score and ROC curve as function of predicted CDS fraction for *A. thaliana* chromosomes. The known CDS fraction of the *A. thaliana* genome is 0.32[1]. Shown on the ROC curve are some values of predicted CDSF which lie close to the known coding fraction.

3.0.14 Random Forest: G3PO Dataset

Accuracy on the G3PO dataset was impacted by the lack of estimated DNA coding fractions. Nevertheless, good results were obtained when compared to other leading methods. Many experimental modifications to the method have been made which were determined to be either good or bad depending on the prediction results on this dataset. Some of these results are presented in this section.

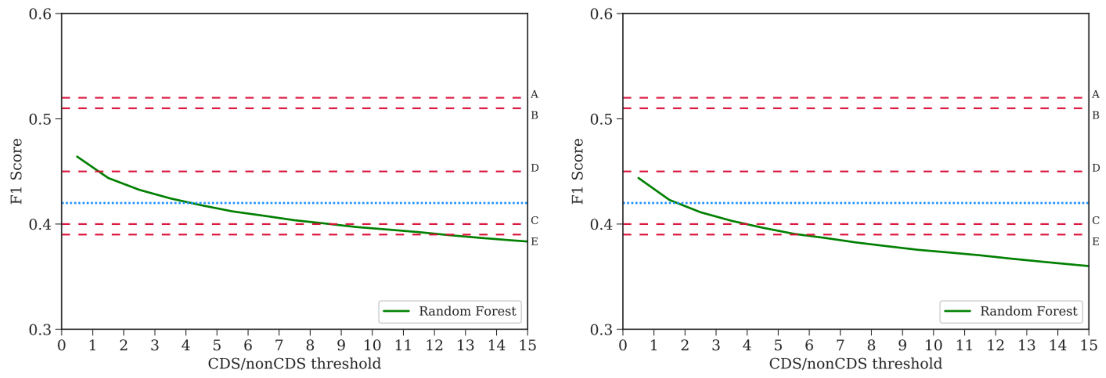


Figure 3.18: A comparison of mean F1 scores when using different methods of setting a maximum CDS length to consider. The left figure shows results from a method which counts the number of downstream stop/donor motifs. The right figure uses a method which sets the maximum number of nts. A clear increase in F1 score can be seen in the method which counts stop/donor motifs. Note that this result uses the first definition of CDS/nonCDS threshold.

Alternate methods of setting the maximum length of potential CDS regions were compared. Across many trials, the method which counts the number of GT or stop codon sites was found to be superior. A figure is shown in this section which indicates measurable drops in F1 score on the G3PO dataset when using a maximum nt method versus maximum number of GT or stop codon motif sites.

All the results shown in this section were generated using RF classifiers trained on 100 thousand samples with a 50/50 split of positive and negative samples. It was found that increases in the number of training samples gave virtually no benefit. In the figure below, Sn, PPV, Sp, BA and F1 scores are compared between RFs which were trained on reduced training sets. It is easily seen that the RFs which were trained on larger amounts of data (100K samples) barely perform any better than those trained on less data. This is a possible advantage as small amounts of training data could potentially be incorporated from non-model organisms.

Table 3.6: Comparison of Sn, PPV, Sp, BA and F1 values on the G3PO dataset between RFs trained on 25, 50 and 100 thousand sample nts. A threshold of 0.9 was chosen because it was found to give good mean accuracy over the whole dataset.

Samples	Sn	PPV	Sp	BA	F1	Th
25K	0.90	0.41	0.16	0.53	0.48	0.9
50K	0.90	0.41	0.16	0.53	0.49	0.9
100K	0.89	0.41	0.17	0.53	0.49	0.9

Below is a table of values which include published results from five leading methods as well as results from the RF based method with several threshold settings. Raw results from the RF indicate the accuracy of predictions before the CDS prediction algorithm has been applied. As before, the fixed threshold indicates means results taken for a single fixed threshold value. F1, BA and JBA rows give the mean values of results when the thresholds are slected to maximize the F1, BA and JBA scores of each gene species individually. The JBA metric is an experimental metric we created which is equal to:

$$JBA = BA - |Sn - Sp| \tag{3.1}$$

From table 3.7, it can be seen that a fixed threshold of 0.9 gives an F1 score of 0.49. At this threshold the RF based method defeats methods GeneID, GlimmerHMM and Snap but falls behind AUGUSTUS and Genscan in terms of F1 score by 0.03 and 0.02 respectively. When allowed to select thresholds which maximize F1 score, giving an upper bound estimate, the RF based method is capable of defeating all five competing methods with an F1 score of 0.55.

Table 3.7: Comparison of Sn, PPV, Sp, BA and F1 values on the G3PO dataset when CDS/nonCDS threshold is chosen to maximize various metrics.

	Sn	PPV	Sp	BA	F1	Th	SD
AUGUSTUS	0.51	0.58	NR	NR	0.52	NA	NA
Genscan	0.50	0.57	NR	NR	0.51	NA	NA
GeneID	0.38	0.52	NR	NR	0.40	NA	NA
GlimmerHMM	0.74	0.43	NR	NR	0.45	NA	NA
Snap	0.38	0.45	NR	NR	0.39	NA	NA
RF - Raw	0.44	0.49	0.74	0.59	0.40	NA	NA
RF - F1	0.73	0.50	0.48	0.61	0.55	1.6	1.9
RF - BA	0.61	0.48	0.71	0.66	0.46	2.1	1.7
RF - JBA	0.58	0.44	0.57	0.58	0.44	1.5	0.7
RF - Fixed	0.79	0.43	0.31	0.55	0.49	0.9	NA

In figures 3.19 and 3.20, it can be seen that mean F1 score and Sn increase as a result of the CDS prediction method. However, mean PPV decreases regardless of the threshold value. This behavior is a strong indication that false positives are an issue even a very high thresholds. The cause of this is currently undetermined.

Low mean PPV values of the method are a remaining weakness of the method. Some things to note are that that this mean is over 1793 genes with a single CDS/nonCDS threshold value. Higher values of both PPV and F1 score, which is related to PPV can be expected when individual thresholds can be set and this can be seen in the value of the mean OOV when the best threshold is selected for each gene in table 3.7. Other methods such as GlimmerHMM and Snap also suffer from low mean PPV on this dataset, while the methods AUGUSTUS, Genscan and GeneID have higher mean PPV values at the expense of lower mean Sn. This trade off can be quantified by the F1 score to some extent. AUGUSTUS and Genscan achieve slightly higher mean F1 scores than the RF method, indicating that the trade off was worth it. However the other methods appear to sacrifice too much Sn for their gain in PPV as indicated by lower mean F1 scores.

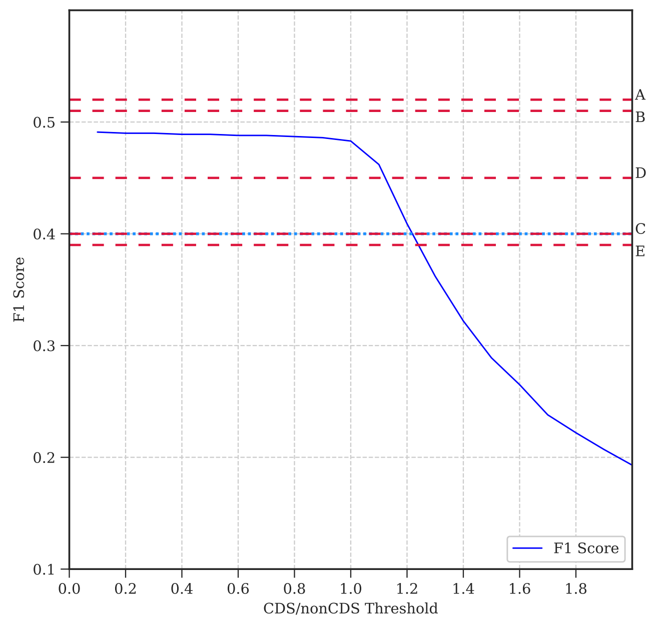


Figure 3.19: Mean random forest F1 scores over G3PO dataset with fixed CDS/nonCDS thresholds. Mean values from competing methods are shown by red dotted lines:: A) AUGUSTUS, B) Genscan, C) GeneID, D) GlimmerHMM, E) Snap. A dotted blue line indicates RF results before CDS prediction algorithm.

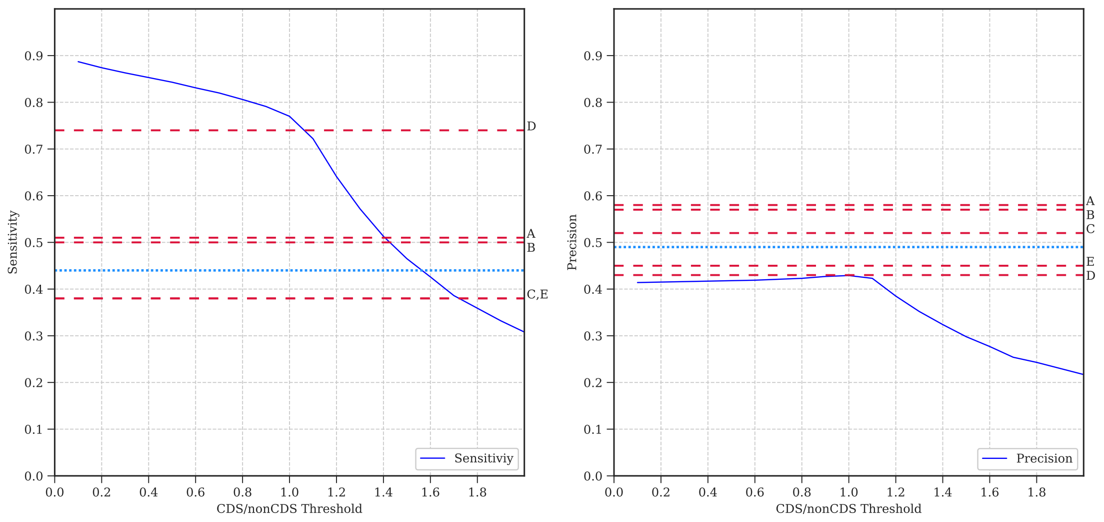


Figure 3.20: Mean random forest sensitivities and precisions over G3PO dataset with fixed CDS/nonCDS thresholds. Mean values from competing methods are shown by red dotted lines: A) AUGUSTUS, B) Genscan, C) GeneID, D) GlimmerHMM, E) Snap. A dotted blue line indicates RF results before CDS prediction algorithm.

3.0.15 Remaining Challenges and Potential Improvements

One of the most difficult problems to solve has been the way in which this method deals with overlapping potential CDS regions. A true CDS region will always have overlapping potential CDS regions due to the regular occurrence of splice site motifs such as YAG and GT which arise by chance and do not actually delineate true CDS boundaries. Start and stop codon motifs also occur by chance.

When the set of all possible CDS regions (potential CDS regions) is generated, each potential CDS can represent one of three possible states. In the first state, a potential CDS region perfectly overlaps a true CDS region. In this case, the boundaries of the potential CDS perfectly locate the splice sites of the true CDS. In the second state, the potential CDS overlaps a true CDS but only partially. The third possibility is that a potential CDS does no overlap a true CDS region at all. The vast majority of potential CDS regions fall into the second two categories.

A NN or RF which is a perfect predictor in terms of nucleotide level classification would create a set of predictions in which potential CDS regions which are actually true CDS regions would be completely filled with positive predictions while potential CDS regions which overlap true CDS regions either partially or not at all. With such a perfect predictor, true CDS regions could be easily separated from false CDS regions because the ratio of true versus false nt level predictions would be much higher in potential CDS regions which perfectly overlap the true CDS regions.

In practice, no NN or RF tested so far is a perfect predictor. This means that true and false CDS regions can not always be separated by comparing the ratio of positive and negative predictions inside each potential CDS region. While precise exon boundaries are not well predicted by the method as of yet, the high values of Sn and Sp at the nucleotide level indicate that the majority of wrong exons are actually overlapping true exons. If this were not the case then Sn and Sp would be lower.

CHAPTER 4: Conclusion

This research project examined the effectiveness of new neural network and random forest based gene prediction methods, aiming to enhance our ability to locate protein coding regions of DNA in uncharacterized sequences. Through comprehensive analysis and experimentation, this work has resulted in the development of methods which can exceed many current methods in terms of accuracy. This was especially clear when used on a large number of non-model organisms.

Neural networks demonstrated their proficiency in learning patterns and extracting meaningful representations from genetic data. With their ability to learn improved representations of data and to explore the altered feature space, neural networks exhibited great potential in gene prediction tasks. Through training on relatively small amounts of data, they acquired deep insights into underlying genetic patterns, enabling accurate gene annotation and classification.

Random forests, known for their ability to handle noisy data and nonlinear relationships, also demonstrated performance in gene prediction tasks which were measurably superior to many existing methods. The ensemble-based approach, leveraging multiple decision trees, allowed for accurate predictions and allowed clear analysis of feature importance. The random forest based method effectively captured unknown patterns within gene sequences and yielded accurate predictions of coding regions.

By comparing the performance of random forests and neural networks, this research project highlighted some differences as well as some shared strengths and weaknesses in gene prediction. Both NN and RF based methods were able to achieve their results on relatively small amounts of training data. The overall results in terms of prediction accuracy were very good compared to other methods in the field. However a consistent weakness of both methods was a tendency towards false positive predictions.

Furthermore, this study emphasized the importance of feature extraction and engineering techniques in improving the performance of both random forests and neural networks. Careful selection and extraction of relevant features played a crucial role in optimizing the predictive models. The modular design of these methods allows for rapid implementation of modifications which included the addition of new sensors. Gene prediction algorithms which can serve as feature extraction sensors are constantly being developed which can be incorporated into this method. Much of the time spent on this project was research into sensors. In addition, the feature space extracted by the sensors is in a format which is easily processed by any machine learning methods.

In conclusion, the research project not only contributed to the advancement of gene prediction methodologies but also provided further evidence for the potential of non-HMM based machine learning algorithms, specifically random forests and neural networks, in deciphering complex biological signals. The findings of this study have implications in various domains, including genetics, bioinformatics, and personalized medicine, ultimately moving us a step closer towards a deeper understanding of the genome and its functional elements.

REFERENCES

[1] S. E. Ahnert, T. M. Fink, and A. Zinovyev, “How much non-coding dna do eukaryotes require?,” *Journal of theoretical biology*, vol. 252, no. 4, pp. 587–592, 2008.

[2] E. R. Mardis, “Dna sequencing technologies: 2006–2016,” *Nature protocols*, vol. 12, no. 2, pp. 213–218, 2017.

[3] S. R. Eddy, “Non–coding rna genes and the modern rna world,” *Nature Reviews Genetics*, vol. 2, no. 12, pp. 919–929, 2001.

[4] J. S. Mattick, P. P. Amaral, P. Carninci, S. Carpenter, H. Y. Chang, L.-L. Chen, R. Chen, C. Dean, M. E. Dinger, K. A. Fitzgerald, *et al.*, “Long non-coding rnas: definitions, functions, challenges and recommendations,” *Nature Reviews Molecular Cell Biology*, vol. 24, no. 6, pp. 430–447, 2023.

[5] F. S. Collins, M. Morgan, and A. Patrinos, “The human genome project: lessons from large-scale biology,” *Science*, vol. 300, no. 5617, pp. 286–290, 2003.

[6] R.-F. Yeh, L. P. Lim, and C. B. Burge, “Computational inference of homologous gene structures in the human genome,” *Genome research*, vol. 11, no. 5, pp. 803–816, 2001.

[7] E. Birney, M. Clamp, and R. Durbin, “Genewise and genomewise,” *Genome research*, vol. 14, no. 5, pp. 988–995, 2004.

[8] M. Stanke, O. Schöffmann, B. Morgenstern, and S. Waack, “Gene prediction in eukaryotes with a generalized hidden markov model that uses hints from external sources,” *BMC bioinformatics*, vol. 7, no. 1, pp. 1–11, 2006.

[9] N. J. Dimonaco, W. Aubrey, K. Kenobi, A. Clare, and C. J. Creevey, “No one tool to rule them all: prokaryotic gene prediction tool annotations are highly dependent on the organism of study,” *Bioinformatics*, vol. 38, no. 5, pp. 1198–1207, 2022.

[10] M. Stanke and S. Waack, “Gene prediction with a hidden markov model and a new intron submodel,” *Bioinformatics*, vol. 19, no. suppl_2, pp. ii215–ii225, 2003.

[11] M. Ahmad, L. T. Jung, and A.-A. Bhuiyan, “From dna to protein: Why genetic code context of nucleotides for dna signal processing? a review,” *Biomedical Signal Processing and Control*, vol. 34, pp. 44–63, 2017.

[12] M. Akhtar, E. Ambikairajah, and J. Epps, “Digital signal processing techniques for gene finding in eukaryotes,” in *Image and Signal Processing: 3rd International Conference, ICISP 2008. Cherbourg-Octeville, France, July 1-3, 2008. Proceedings 3*, pp. 144–152, Springer, 2008.

[13] M. Bauer, S. M. Schuster, and K. Sayood, "The average mutual information profile as a genomic signature," *BMC bioinformatics*, vol. 9, no. 1, pp. 1–11, 2008.

[14] M. J. Berryman, A. Allison, and D. Abbott, "Mutual information for examining correlations in dna," *Fluctuation and Noise Letters*, vol. 4, no. 02, pp. L237–L246, 2004.

[15] I. Grosse, S. V. Buldyrev, H. E. Stanley, D. Holste, and H. Herzel, "Average mutual information of coding and noncoding dna," in *Biocomputing 2000*, pp. 614–623, World Scientific, 1999.

[16] M. Dehnert, W. E. Helm, and M.-T. Hütt, "Information theory reveals large-scale synchronisation of statistical correlations in eukaryote genomes," *Gene*, vol. 345, no. 1, pp. 81–90, 2005.

[17] G. Dodin, P. Vandergheynst, P. Levoir, C. Cordier, and L. Marcourt, "Fourier and wavelet transform analysis, a tool for visualising regular patterns in dna," *Journal of Theoretical Biology*, vol. 206, no. ARTICLE, pp. 323–326, 2000.

[18] Q. Zheng, T. Chen, W. Zhou, L. Xie, and H. Su, "Gene prediction by the noise-assisted memd and wavelet transform for identifying the protein coding regions," *Biocybernetics and Biomedical Engineering*, vol. 41, no. 1, pp. 196–210, 2021.

[19] M. K. Hota and V. K. Srivastava, "Performance analysis of different dna to numerical mapping techniques for identification of protein coding regions using tapered window based short-time discrete fourier transform," in *2010 International Conference on Power, Control and Embedded Systems*, pp. 1–4, IEEE, 2010.

[20] D. Kotlar and Y. Lavner, "Gene prediction by spectral rotation measure: a new method for identifying protein-coding regions," *Genome research*, vol. 13, no. 8, pp. 1930–1937, 2003.

[21] T. W. Fox and A. Carreira, "A digital signal processing method for gene prediction with improved noise suppression," *EURASIP Journal on Advances in Signal Processing*, vol. 2004, pp. 1–7, 2004.

[22] M. Mabrouk, "Advanced genomic signal processing methods in dna mapping schemes for gene prediction using digital filters," *American Journal of Signal Processing*, vol. 7, no. 1, pp. 12–24, 2017.

[23] S. Singha Roy and S. Barman, "Polyphase filtering with variable mapping rule in protein coding region prediction," *Microsystem Technologies*, vol. 23, pp. 4111–4121, 2017.

[24] P. Vaidyanathan and B.-J. Yoon, "Digital filters for gene prediction applications," in *Conference Record of the Thirty-Sixth Asilomar Conference on Signals, Systems and Computers, 2002.*, vol. 1, pp. 306–310, IEEE, 2002.

[25] J. Gao, Y. Qi, Y. Cao, and W.-w. Tung, "Protein coding sequence identification by simultaneously characterizing the periodic and random features of dna sequences," *Journal of biomedicine and biotechnology*, vol. 2005, no. 2, p. 139, 2005.

[26] S. Buldyrev, N. Dokholyan, A. Goldberger, S. Havlin, C.-K. Peng, H. Stanley, and G. Viswanathan, "Analysis of dna sequences using methods of statistical physics," *Physica A: Statistical Mechanics and its Applications*, vol. 249, no. 1-4, pp. 430–438, 1998.

[27] S. Ossadnik, S. Buldyrev, A. Goldberger, S. Havlin, R. Mantegna, C. Peng, M. Simons, and H. Stanley, "Correlation approach to identify coding regions in dna sequences," *Biophysical Journal*, vol. 67, no. 1, pp. 64–70, 1994.

[28] Y. Xu, R. J. Mural, J. R. Einstein, M. B. Shah, and E. C. Uberbacher, "Grail: a multi-agent neural network system for gene identification," *Proceedings of the IEEE*, vol. 84, no. 10, pp. 1544–1552, 1996.

[29] R. Silva, K. Padovani, F. Góes, and R. C. Alves, "A random forest classifier for prokaryotes gene prediction," in *2019 8th Brazilian conference on intelligent systems (BRACIS)*, pp. 545–550, IEEE, 2019.

[30] A. Al-Ajlan and A. El Allali, "Feature selection for gene prediction in metagenomic fragments," *BioData mining*, vol. 11, no. 1, pp. 1–12, 2018.

[31] R. Damaševicius, "Splice site recognition in dna sequences using k-mer frequency based mapping for support vector machine with power series kernel," in *2008 International Conference on Complex, Intelligent and Software Intensive Systems*, pp. 687–692, IEEE, 2008.

[32] N. Goel, S. Singh, and T. C. Aseri, "A comparative analysis of soft computing techniques for gene prediction," *Analytical biochemistry*, vol. 438, no. 1, pp. 14–21, 2013.

[33] D. K. D. Haussler and M. Eeckman, "A generalized hidden markov model for the recognition of human genes in dna," in *Proc. int. conf. on intelligent systems for molecular biology, st. louis*, pp. 134–142, 1996.

[34] A. Lomsadze, V. Ter-Hovhannisyan, Y. O. Chernoff, and M. Borodovsky, "Gene identification in novel eukaryotic genomes by self-training algorithm," *Nucleic acids research*, vol. 33, no. 20, pp. 6494–6506, 2005.

[35] K. Anders and M. Kasper, "Automatic generation of gene finders for eukaryotic species," *BMC Bioinformatics*, 2006.

[36] W. H. Majoros, M. Pertea, and S. L. Salzberg, "Tigrscan and glimmerhmm: two open source ab initio eukaryotic gene-finders," *Bioinformatics*, vol. 20, no. 16, pp. 2878–2879, 2004.

[37] N. Scalzitti, A. Jeannin-Girardon, P. Collet, O. Poch, and J. D. Thompson, "A benchmark study of ab initio gene prediction methods in diverse eukaryotic organisms," *BMC genomics*, vol. 21, no. 1, pp. 1–20, 2020.

[38] R. Guigó, P. Flicek, J. F. Abril, A. Reymond, J. Lagarde, F. Denoeud, S. Antonarakis, M. Ashburner, V. B. Bajic, E. Birney, *et al.*, "Egasp: the human encode genome annotation assessment project," *Genome biology*, vol. 7, no. 1, pp. 1–31, 2006.

[39] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, pp. 115–133, 1943.

[40] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain.," *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[41] A. El Allali and J. R. Rose, "Mgc: a metagenomic gene caller," *BMC bioinformatics*, vol. 14, pp. 1–10, 2013.

[42] M. R. Amin, A. Yurovsky, Y. Tian, and S. Skiena, "Deepannotator: genome annotation with deep learning," in *Proceedings of the 2018 ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, pp. 254–259, 2018.

[43] K. J. Hoff, M. Tech, T. Lingner, R. Daniel, B. Morgenstern, and P. Meinicke, "Gene prediction in metagenomic fragments: a large scale machine learning approach," *BMC bioinformatics*, vol. 9, pp. 1–14, 2008.

[44] E. E. Snyder and G. D. Stormo, "Identification of coding regions in genomic dna sequences: an application of dynamic programming and neural networks," *Nucleic acids research*, vol. 21, no. 3, pp. 607–613, 1993.

[45] C. H. Wu, "Artificial neural networks for molecular sequence analysis," *Computers & chemistry*, vol. 21, no. 4, pp. 237–256, 1997.

[46] L. Breiman, "Random forests," *Machine learning*, vol. 45, pp. 5–32, 2001.

[47] M. Burset and R. Guigo, "Evaluation of gene structure prediction programs," *genomics*, vol. 34, no. 3, pp. 353–367, 1996.

[48] E. C. Uberbacher and R. J. Mural, "Locating protein-coding regions in human dna sequences by a multiple sensor-neural network approach," *Proceedings of the National Academy of Sciences*, vol. 88, no. 24, pp. 11261–11265, 1991.

[49] L. Baker, C. David, and D. J. Jacobs, "Ab initio gene prediction for protein-coding regions," *Bioinformatics Advances*, vol. 3, no. 1, p. vbad105, 2023.

[50] J. W. Fickett, "Recognition of protein coding regions in dna sequences," *Nucleic acids research*, vol. 10, no. 17, pp. 5303–5318, 1982.

[51] C. Yin and S. S.-T. Yau, “Numerical representation of dna sequences based on genetic code context and its applications in periodicity analysis of genomes,” in *2008 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, pp. 223–227, IEEE, 2008.

[52] L. Das, J. Das, and S. Nanda, “Identification of exon location applying kaiser window and dft techniques,” in *2017 2nd International Conference for Convergence in Technology (I2CT)*, pp. 211–216, IEEE, 2017.

[53] J.-M. Claverie, I. Sauvaget, and L. Bougueleret, *[15] k-tuple frequency analysis: From intron/exon discrimination to T-cell epitope mapping*, vol. 183 of *Methods in Enzymology.* San Diego :: Academic Press,, 1990.

[54] R. Staden and A. McLachian, “Codon preference and its use in identifying protein coding regions in long dna sequences,” *Nucleic acids research*, vol. 10, no. 1, pp. 141–156, 1982.

[55] J. L. Oliver and A. Marín, “A relationship between gc content and coding-sequence length,” *Journal of Molecular Evolution*, vol. 43, no. 3, pp. 216–223, 1996.

[56] M. Amit, M. Donyo, D. Hollander, A. Goren, E. Kim, S. Gelfman, G. Lev-Maor, D. Burstein, S. Schwartz, B. Postolsky, *et al.*, “Differential gc content between exons and introns establishes distinct strategies of splice-site recognition,” *Cell reports*, vol. 1, no. 5, pp. 543–556, 2012.

[57] K. McNair, C. L. Ecale Zhou, B. Souza, S. Malfatti, and R. A. Edwards, “Utilizing amino acid composition and entropy of potential open reading frames to identify protein-coding genes,” *Microorganisms*, vol. 9, no. 1, p. 129, 2021.

[58] Z. Ouyang, H. Zhu, J. Wang, and Z.-s. She, “Multivariate entropy distance method for prokaryotic gene identification,” *Journal of Bioinformatics and Computational Biology*, vol. 2, no. 02, pp. 353–373, 2004.

[59] R. P. Simões, I. R. Wolf, B. A. Correa, and G. T. Valente, “Uncovering patterns of the evolution of genomic sequence entropy and complexity,” *Molecular Genetics and Genomics*, vol. 296, no. 2, pp. 289–298, 2021.

[60] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, “Self-normalizing neural networks,” *Advances in neural information processing systems*, vol. 30, 2017.

[61] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

[62] M. Burset, I. A. Seledtsov, and V. V. Solovyev, “Analysis of canonical and non-canonical splice sites in mammalian genomes,” *Nucleic acids research*, vol. 28, no. 21, pp. 4364–4375, 2000.

[63] K. Frey and B. Pucker, “Animal, fungi, and plant genome sequences harbor different non-canonical splice sites,” *Cells*, vol. 9, no. 2, p. 458, 2020.

[64] G. E. Parada, R. Munita, C. A. Cerda, and K. Gysling, “A comprehensive survey of non-canonical splice sites in the human transcriptome,” *Nucleic acids research*, vol. 42, no. 16, pp. 10564–10578, 2014.

[65] J. D. Hawkin, “A survey on intron and exon lengths,” *Nucleic acids research*, vol. 16, no. 21, pp. 9893–9908, 1988.

[66] C. Burge and S. Karlin, “Prediction of complete gene structures in human genomic dna,” *Journal of molecular biology*, vol. 268, no. 1, pp. 78–94, 1997.

[67] S. L. Salzberg, M. Pertea, A. L. Delcher, M. J. Gardner, and H. Tettelin, “Interpolated markov models for eukaryotic gene finding,” *Genomics*, vol. 59, no. 1, pp. 24–31, 1999.

[68] R. Guigó, S. Knudsen, N. Drake, and T. Smith, “Prediction of gene structure,” *Journal of molecular biology*, vol. 226, no. 1, pp. 141–157, 1992.

[69] I. Korf, “Gene finding in novel genomes,” *BMC bioinformatics*, vol. 5, no. 1, pp. 1–9, 2004.

[70] J. Brabec, T. Komárek, V. Franc, and L. Machlica, “On model evaluation under non-constant class imbalance,” in *Computational Science–ICCS 2020: 20th International Conference, Amsterdam, The Netherlands, June 3–5, 2020, Proceedings, Part IV 20*, pp. 74–87, Springer, 2020.